EPSON
EXCEED YOUR VISION

ePOS-Device SDK for Android User's Manual

Overview

Describes the features and development environment.

Building Environment

Describes environment building for ePOS-Device Service I/F.

Programming Guide

Describes how to write programs in application development.

API Reference

Describes the APIs provided in ePOS-Device SDK for Android.

Sample Program

Describes the sample program.

Appendix

Describes the list of keycode and printer specifications.

M00077701
Rev.B

Cautions

- No part of this document may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of Seiko Epson Corporation.
- The contents of this document are subject to change without notice. Please contact us for the latest information.
- While every precaution has taken in the preparation of this document, Seiko Epson Corporation assumes no responsibility for errors or omissions.
- Neither is any liability assumed for damages resulting from the use of the information contained herein.
- Neither Seiko Epson Corporation nor its affiliates shall be liable to the purchaser of this product or third parties for damages, losses, costs, or expenses incurred by the purchaser or third parties as a result of: accident, misuse, or abuse of this product or unauthorized modifications, repairs, or alterations to this product, or (excluding the U.S.) failure to strictly comply with Seiko Epson Corporation's operating and maintenance instructions.
- Seiko Epson Corporation shall not be liable against any damages or problems arising from the use of any options or any consumable products other than those designated as Original EPSON Products or EPSON Approved Products by Seiko Epson Corporation.

Trademarks

EPSON is a registered trademark of Seiko Epson Corporation.

Exceed Your Vision and ESC/POS are registered trademarks or trademarks of Seiko Epson Corporation.

Android™ is either registered trademarks or trademarks of Google Inc. in the United States and other countries.

Java™ is a registered trademark of Oracle Corporation, its subsidiaries, and affiliates in the U.S. and other countries.

Eclipse® is a trademark or registered trademark of Eclipse Foundation, Inc.

Wi-Fi® is a registered trademark of the Wi-Fi Alliance®.

All other trademarks are the property of their respective owners and used for identification purpose only.

ESC/POS® Command System

EPSON has been taking industry's initiatives with its own POS printer command system (ESC/POS). ESC/POS has a large number of commands including patented ones. Its high scalability enables users to build versatile POS systems. The system is compatible with all types of EPSON POS printers (excluding the TM-C100) and displays. Moreover, its flexibility makes it easy to upgrade the future. The functionality and the user-friendliness is valued around the world.

© Seiko Epson Corporation 2014. All rights reserved.

For Safety

Key to Symbols

The symbols in this manual are identified by their level of importance, as defined below. Read the following carefully before handling the product.

	Provides information that must be observed to avoid damage to your equipment or a malfunction.
	Provides important information and useful tips.

Restriction of Use

When this product is used for applications requiring high reliability/safety such as transportation devices related to aviation, rail, marine, automotive etc.; disaster prevention devices; various safety devices etc; or functional/precision devices etc, you should use this product only after giving consideration to including fail-safes and redundancies into your design to maintain safety and total system reliability. Because this product was not intended for use in applications requiring extremely high reliability/safety such as aerospace equipment, main communication equipment, nuclear power control equipment, or medical equipment related to direct medical care etc, please make your own judgment on this product's suitability after a full evaluation.

About this Manual

Aim of the Manual

This manual is intended to provide development engineers with all the information necessary for building/designing an ePOS-Device system or developing/designing an ePOS-Device application, by using TM-i.

The TM-i series in this manual is a generic term for the following printers.

- TM-T20II-i
- TM-T70-i
- TM-T82II-i
- TM-T83II-i
- TM-T88V-i

Manual Content

The manual is made up of the following sections:

Chapter 1	Overview
Chapter 2	Building Environment
Chapter 3	Programming Guide
Chapter 4	API Reference
Chapter 5	Sample Program
Appendix	List of KeyCode
	Printer Specifications
	Open Source License

Contents

■ **For Safety** **3**

Key to Symbols 3

■ **Restriction of Use** **3**

■ **About this Manual** **4**

Aim of the Manual 4
Manual Content 4

■ **Contents** **5**

Overview 11

■ **ePOS-Device SDK** **11**

Features 11
System Configuration Example 12
How to Use the Communication Box 13
Specifying the Print Job ID from the Application 16
Spooler Function 17

■ **Operating Environment** **20**

Android Version 20
Android Device 20
TM-i 20
TM Printer 21
Supported Peripheral Device 21
Development Environment 22

■ **Contents in the Package** **23**

Package 23
Download 23

■ **Restrictions** **24**

Building Environment 25

■ **Workflow** **25**

■ **Registration of Device** **26**

Programming Guide 27

■ **How to Incorporate the ePOS-Device SDK for Android** **27**

■ **Programming Flow** **29**

If Automatically Reconnecting to the Network 30
Programming flow for the Communication Box 32

■ **Using the Spooler Function** **37**

■ **Event Notification Method** **39**

List of Listener Interface 39

API Reference 43

■ **List of ePOS-Device API 43**

Device Class 43
Display Class 44
Keyboard Class.....44
Printer Class 44
Scanner Class 46
SimpleSerial Class 47
CommBoxManager Class 47
CommBox Class.....47
Log Class 47
EposException Class 47
EposCallbackCode Class.....47

■ **Device Class..... 48**

Device 48
connect.....49
disconnect 50
isConnected 50
createDevice.....51
deleteDevice.....53
getAdmin 54
getLocation 54
getCommBoxManager 54
setReconnectingEventCallback 55
setReconnectEventCallback.....56
setDisconnectEventCallback 57

■ **Display Class 58**

sendData.....58
clearCommandBuffer 58
reset 59
createWindow.....60
destroyWindow.....61
setCurrentWindow.....61
clearWindow.....62
setCursorPosition.....62
moveCursorPosition.....63
setCursorType.....64
addText.....65
addReverseText 66
addMarquee 67
setBlink 68
setBrightness.....68
showClock.....69
addCommand 69
setReceiveEventCallback 70

■ **Keyboard Class 71**

setPrefix.....71
setKeyPressEventCallback.....72
setStringEventCallback.....73

■ **Printer Class 74**

sendData 74
getPrintJobStatus 75
clearCommandBuffer 75
addTextAlign 76
addTextLineSpace 76
addTextRotate 77
addText 78
addTextLang 79
addTextFont 80
addTextSmooth 81
addTextDouble 82
addTextSize 83
addTextStyle 84
addTextPosition 86
addTextVPosition 86
addFeedUnit 87
addFeedLine 87
addFeedPosition 88
addFeed 88
addImage 89
addLogo 91
addBarcode 92
addSymbol 96
addHLine 101
addVLineBegin 102
addVLineEnd 103
addPageBegin 104
addPageEnd 104
addPageArea 105
addPageDirection 106
addPagePosition 107
addPageLine 108
addPageRectangle 109
addCut 110
addPulse 111
addSound 112
addLayout 114
addCommand 117
addRecovery 117
addReset 118
startMonitor 118
stopMonitor 118
recover 119
reset 119
halftone property 120
brightness property 121
force property 122
timeout property 123
interval property 123
setReceiveEventCallback 124
setStatusChangeEventCallback 128
setBatteryStatusChangeEventCallback 129

setOnlineEventCallback 130
setOfflineEventCallback 131
setPowerOffEventCallback 132
setCoverOkEventCallback 133
setCoverOpenEventCallback 134
setPaperOkEventCallback 135
setPaperNearEndEventCallback 136
setPaperEndEventCallback 137
setDrawerClosedEventCallback 138
setDrawerOpenEventCallback 139
setBatteryOkEventCallback 140
setBatteryLowEventCallback 141

■ **Scanner Class 142**

setDataEventCallback 142

■ **SimpleSerial Class 143**

sendCommand 143
setCommandReplyEventCallback 144

■ **CommBoxManager Class 145**

openCommBox 145
closeCommBox 147

■ **CommBox Class 149**

getCommHistory 149
sendData 151
setReceiveEventCallback 153

■ **EposDeviceLog Class 154**

setLogSettings 154

■ **EposException class 156**

getErrorStatus 156

Sample Program 157

■ **Functionality 157**

■ **Usage Environment 158**

Development Environment 158
Required device 158
Android Device 158

■ **Environment Setting 159**

Setting the Devices 160
Installing the Sample Program 161

■ **How to Use the Sample Program 162**

Setting 162
Operation 164

Appendix 165

■ **List of KeyCode** **165**

■ **Printer Specifications** **166**

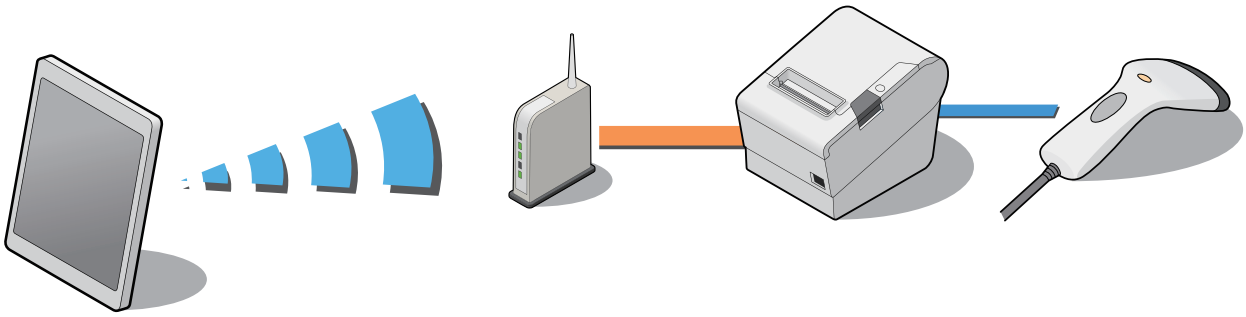
TM-T20II-i 166
TM-T70-i 168
TM-T82II-i 169
TM-T83II-i/ TM-T83II 170
TM-T88V-i/ TM-T88V 172
TM-L90 174
TM-P20 176
TM-P60II 178
TM-P60II with Peeler 179
TM-P80 180
TM-T20 181
TM-T20II 182
TM-T70 183
TM-T70II 184
TM-T82II 185
TM-T88IV 186
TM-T90 187
TM-U220 189

■ **Open Source License** **190**

Overview

This chapter describes the features of and the specifications for ePOS-Device SDK for Android.

ePOS-Device SDK



The ePOS-Device SDK for Android is SDK for developers to develop Android application that controls various peripheral devices connected to TM-i. Application can be developed with API provided by the ePOS-Device SDK.

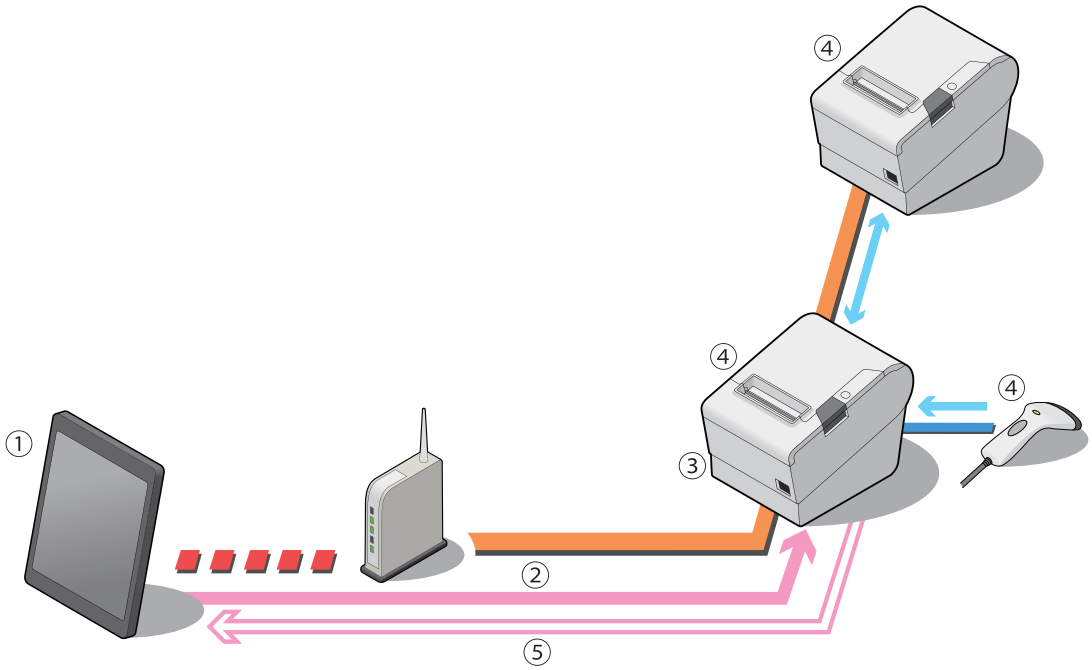
The ePOS-Device SDK also has the "ePOS-Device SDK for iOS" for iOS applications.

Features

- ❏ Able to use in any environment that network connection is available.
- ❏ No need to prepare any device (such as a PC) to act as a controller to control the peripheral devices.
- ❏ Accessing a device using ePOS-Device SDK automatically locks the device exclusively.
  Even if accessed from multiple terminals simultaneously, the device is not under multiple controls. When the terminal that has controlled the device releases the device, the device becomes controllable from another terminal.
- ❏ Data can be sent and received between applications.
  For details, refer to How to Use the Communication Box(p.13).
- ❏ Printing requests can be made to the printer by specifying the printing job ID. *
  For details, refer to Specifying the Print Job ID from the Application(p.16).
- ❏ Printing data can be sent to the printer without waiting for a printing complete response from the printer. (Spooler) * For details, refer to Spooler Function(p.17).
- ❏ Print data can be sent to another printer if the printer sends an error response (PAPER END, COVER OPEN, etc.) or does not return a response. (Print forwarding) *
  For details, refer to Forwarding(p.19).

* TM-i firmware Ver.4.1 or later versions supported

System Configuration Example

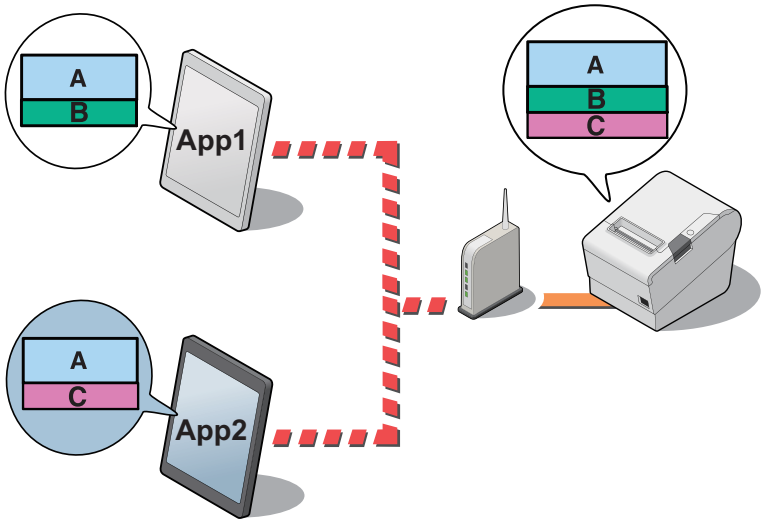


1. Place an Android application in the Android device.
2. The Android application sends a request message to TM-i.
3. TM-i sends data to devices that can be controlled.
4. The devices and network printers connected to TM-i are controlled.
5. TM-i returns a response to the application.

How to Use the Communication Box

The Communication Box is a virtual data space that is provided by the ePOS-Device Service I/F for sending and receiving data between applications.

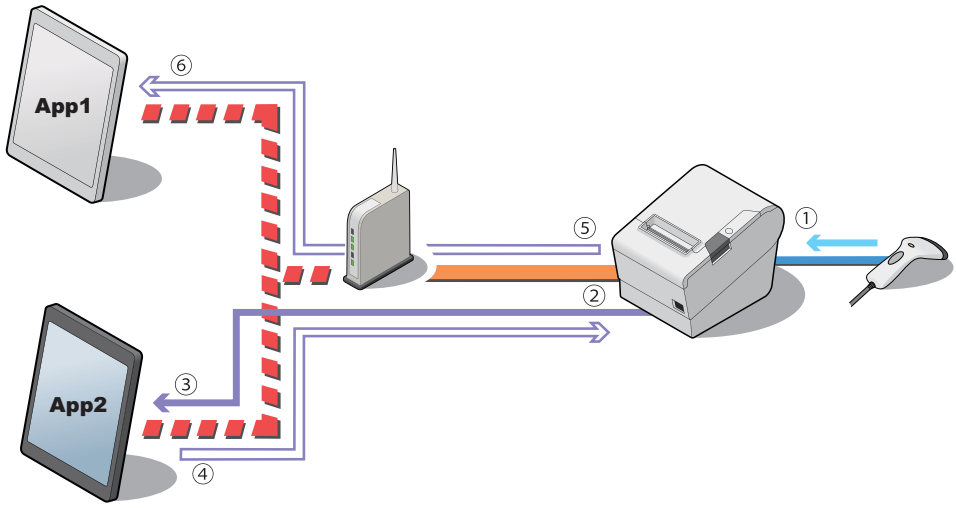
> The ePOS-Device is providing a virtual space and structure for sending and receiving data between applications. Create the data to be sent and received according to your application's specifications.



The Communication Box can be used in the following ways.

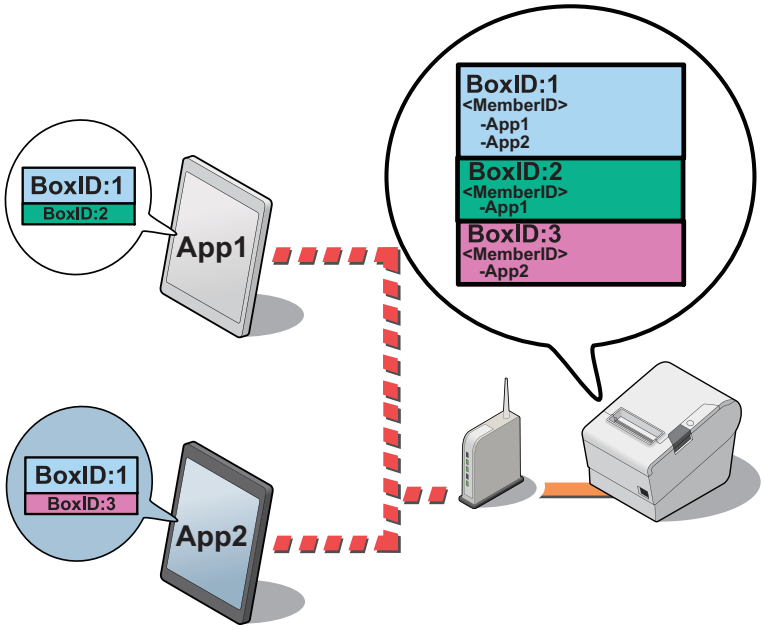
- Using an Android device as a POS display terminal
- Displaying digital signage
- Using a tablet terminal as an entry terminal

Example of data processing using the Communication Box



1. The TM-i receives the barcode data scanned by the scanner.
2. The ePOS-Device Service I/F notifies App2 of the barcode data.
3. App2 acquires barcode data and converts to POS data.
4. App2 sends the POS data to the ePOS-Device Service I/F's Communication Box.
5. The ePOS-Device Service I/F notifies App1 that the POS data is stored in the Communication Box.
6. App1 acquires the POS data stored in the Communication Box.

Structure of the Communication Box



The ePOS-Device Service I/F manages Communication Boxes with Box IDs (BoxID in the figure above).

Data can be sent and received between applications belonging to the Communication Box.

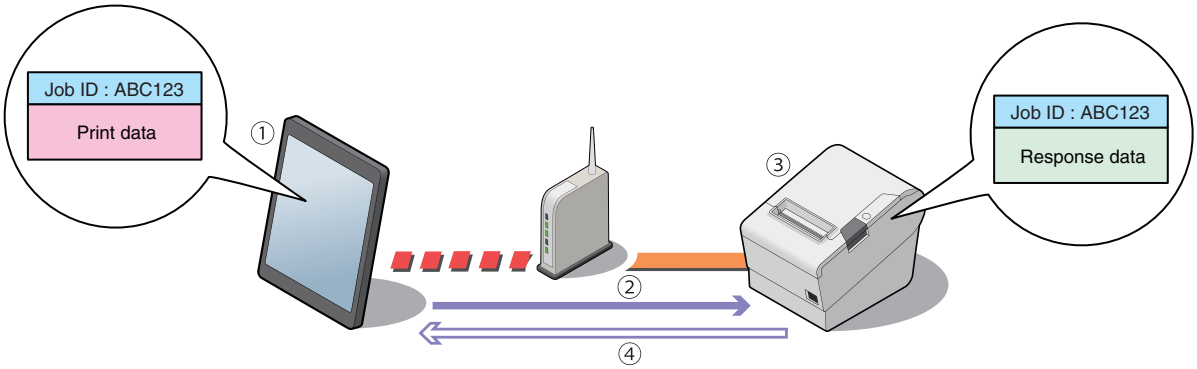
In the figure above, applications App1 and App2 can send and receive data with one another using the BoxID:1 Communication Box.

Specifications of the Communication Box

Maximum number of Communication Boxes that can be created	20
Maximum number of applications that can belong to one Communication Box	20
Space of transfer history that one Communication Box can maintain	10240 Byte
Size of data that can be sent at one time	1024 Byte

Specifying the Print Job ID from the Application

A response containing the specified print job ID will be returned when sending a request from the application by specifying the print job ID. (TM-i firmware Ver.4.1 or later versions)



Spooler Function

The spooler function is a function used to temporarily store printing data received from the application in the TM-i series printer to carry out background printings sequentially. (TM-i firmware Ver.4.1 or later versions)

TM-i series printers return response data to the application the moment the printing data enters the spooler to make it possible for applications to proceed to the next process without waiting for a printing complete response.

The application can use the job ID included in the response data to send an inquiry on the printing status.

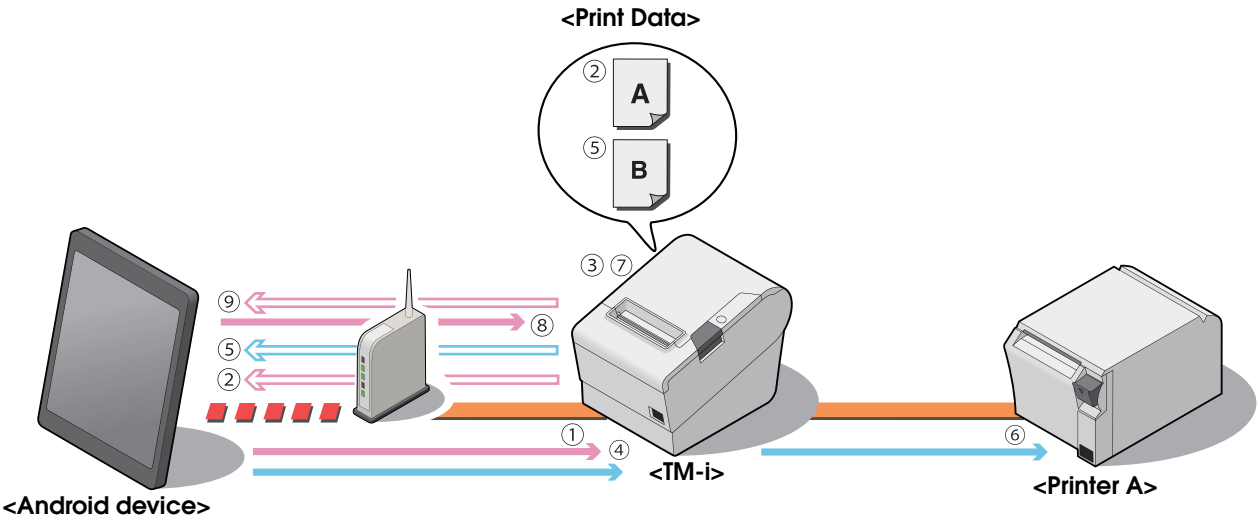
EPSON TMNet WebConfig is used to configure the spooler.

Perform the following settings. For details, refer to the Technical Reference Guide for each TM-i.

- Settings to enable the spooler
- Print process retry settings

Forwarding can be used to send a printing job to a different printer if the printer does not respond to retry processing. For details, refer to Forwarding(p.19).

Example



1. Send print data A to a TM-i (ePOS-Device Service I/F) from the Android device.
2. The TM-i will save the print data A to the spooler and will return response data to the Android device.
3. The TM-i will retrieve the print data and issue a print command to another TM-i.

4 Print data B will be sent to the TM-i from the input terminal.

5 The TM-i will save the print data B to the spooler and will return response data to the Android device.

6 The TM-i will retrieve the print data B and issue a print command to printer A.

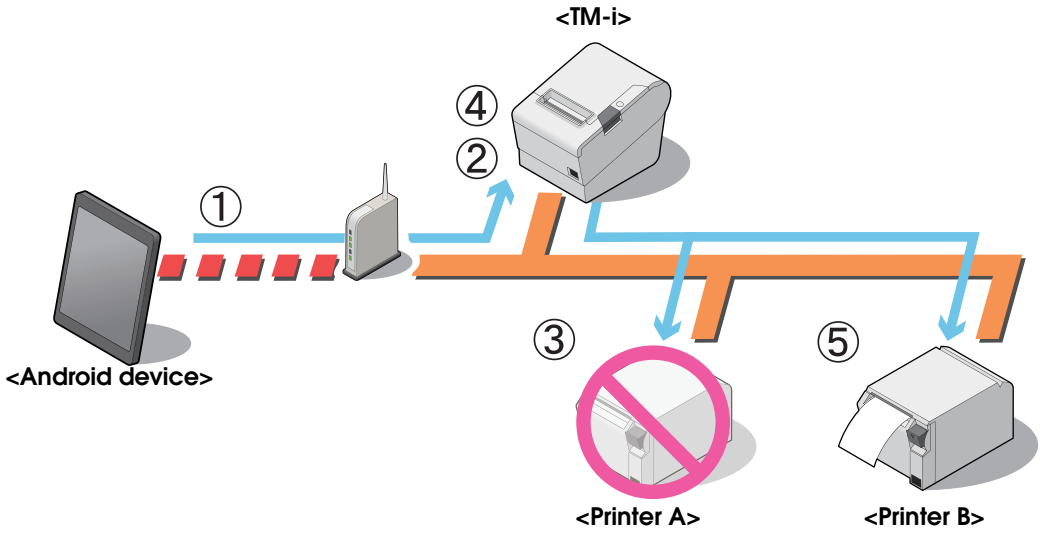
7 Print data A is complete.

8 A query on print data A results will be sent to the TM-i from the Android device.

9 The TM-i will return response data back to the Android device signifying that print data A completed without error.

Forwarding

EPSON TMNet WebConfig is used to configure forwarding settings. For details, refer to the Technical Reference Guide for each TM-i.



1. Send print data to ePOS-Device Service I/F from the Android device.
2. The TM-i issues a print command to printer A.
3. Printer A does not respond.
4. The TM-i reissues a print command to printer A.
   If printer A still does not respond to the reissued print command, the print command will be issued to printer B.
5. Print data sent from the Android device will be printed using printer B.

Operating Environment

Android Version

- ❏ Android Version 2.3.3 to 2.3.7
- ❏ Android Version 3.1 to 3.2.2
- ❏ Android Version 4.0 or later

For the latest version, refer to the README file.

Android Device

- ❏ Device that supports ARMv5TE

TM-i

- ❏ TM-T20II-i
- ❏ TM-T70-i (TM-i Firmware Ver.4.0 or later)
- ❏ TM-T82II-i
- ❏ TM-T83II-i
- ❏ TM-T88V-i (TM-i Firmware Ver.4.0 or later)

TM Printer

In addition to the TM-i local printer, the following TM printers can be controlled:

TM Printer	TM-i Firmware	
	Ver.4.0	Ver.4.1
TM-L90*	✔	✔
TM-P20	-	✔
TM-P60II	✔	✔
TM-P60II Peeler	✔	✔
TM-P80	✔	✔
TM-T20*	✔	✔
TM-T20II*	✔	✔
TM-T70*	✔	✔
TM-T70II*	✔	✔
TM-T82II*	✔	✔
TM-T83II*	-	✔
TM-T88IV*	✔	✔
TM-T88V*	✔	✔
TM-T90*	✔	✔
TM-U220*	✔	✔

* Ethernet model (UB-E02/ UB-E03), or wireless LAN model (UB-R03/ UB-R04).
Other interface models are not supported.

Interface

Use a TM printer with the following interface:

- ❏ Wired LAN
- ❏ Wireless LAN (UB-R03/ UB-R04)

Supported Peripheral Device

To use a peripheral device, the user must prepare a device control script file developed for the device. An API for the device control script is also provided. For details, refer to Sample Program(p.157).

Peripheral Device

- ❏ Customer display
  - DM-D110 USB Interface
- ❏ Standard HID device
  - Keyboard device
  - Barcode scanner
- ❏ Serial communication device

When controlling a serial communication device with this SDK, API to send/receive commands is used to control.

Development Environment

The following are necessary to develop an Android application.

- Android SDK r15 or later
- Java Development Kit 6 or later

Contents in the Package

Package

File	Description
ePOS-Device.jar	Compiled Java class file, archived into a jar format file to allow APIs to be used from Java programs.
libeposdevice.so	Library for function execution. (ARMv5TE supported)
ePOS-Device_Sample_Android.zip	A sample program file.
README.en.txt	A readme file.
EULA.en.txt	Contains the SOFTWARE LICENSE AGREEMENT.
ePOS-Device_SDK_Android_en_revx.pdf	This manual. Describes the ePOS-Device SDK for Android programming methods and APIs.

Download

For customers in North America, go to the following web site:

http://www.epsonexpert.com/

For customers in other countries, go to the following web site:

https://download.epson-biz.com/?service=pos/

Restrictions

- ❏ The ruled line command is available only when the printer supports the "ruled line command".
- ❏ The drawer and the buzzer cannot be used together.
- ❏ The buzzer function cannot be used if the printer is not provided with the buzzer.
- ❏ When a 2D-code scanner is used, multibyte characters such as Japanese cannot be obtained properly.
- ❏ When 2D-code data contains an ASCII control code (0x00 to 0x1F), control codes cannot be obtained.
- ❏ Keycodes that can be obtained from the keyboard are restricted. For the key codes that can be obtained, refer to List of KeyCode(p.165).

Building Environment

This chapter describes environment building for ePOS-Device Service I/F.

Workflow

1. Network Settings

Configure the system and TM-i network settings.

Perform TM-i network settings using the following procedure.
For details, refer to the Technical Reference Guide of each TM-i.

- EPSON TMNet WebConfig
- EpsonNet Config
- Easy Kitting

2.Connection of Device to TM-i

Connect a device to TM-i. For the devices that can be connected, refer to Operating Environment (p.20).

3.Registration of Device (p.26)

Register the device to be connected into TM-i. Make registration using an Epson TMNet WebConfig. For details, refer to the Technical Reference Guide for each TM-i.

Registration of Device

Register the device to be controlled by ePOS-Device SDK for Android into TM-i. Configure the settings using EPSON TMNet WebConfig.

EPSON TMNet WebConfig can be launched from your web browser.

For details, refer to the Technical Reference Guide for each TM-i.

Configure the following settings for each device connected to the TM-i.

Printer

Item	Description
Device ID	Enter the ID (any character string) of the printer to be controlled.
Model	Select the model of the printer to be controlled.
IP Address	Specify the IP address of the printer for each device ID.
Retry interval(ms)	Specify the retry interval after timeout.

Customer Display

Item	Description
Customer Display	Set whether or not this is used.
Communications settings	Set the communication speed, data bit and parity.
Brightness settings	Set the brightness of the customer display.

The device ID of the customer display is fixed to "local_display".

Key input device

Item	Description
Device ID	Enter the device ID (any character string).
Device name	Select the device name of the key input device.
Control Script	Select the device control script to be used for the key input device.

Serial communications

Item	Description
Serial communications	Set whether or not this is used.
Communication speed (bps)	Set the communication speed of the device.
Data bit	Set the data bit.
Parity	Set the parity.
Stop bit	Set the stop bit.
Flow control	Set the flow control.

The device ID of the serial communication device is fixed to "local_serial".

Programming Guide

This chapter describes how to write programs in the application development using ePOS-Device SDK for Android.

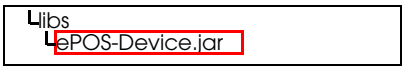
How to Incorporate the ePOS-Device SDK for Android

This section explains how to incorporate the ePOS-Device SDK for Android.

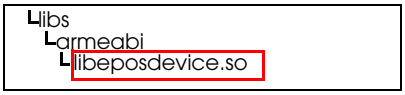
> This explanation uses Eclipse. If you are using another development environment, please make the appropriate changes.

Incorporate the SDK using following procedures.

1. Create a new project in Eclipse.
2. Copy provided JAR file (ePOS-Device.jar) into following path:



3. In Libraries tab of the target project's properties, confirm that the JAR file you added (ePOS-Device.jar) is registered in (Java Build Path).
   If it has not been added, add the JAR file into build path using (Add Jars...).
4. Copy the library file (libeposdevice.so) into following path:



5. Select the project in Eclipse's Package Explorer, right click on it, and press (Refresh).
6. In *.java source file of application to use, enter package import definitions per class. Refer to the following.

Class	Import Definition
Device class	`import com.epson.eposdevice;`
Log class	
EposException class	
EposCallbackCode class	
Display class	`import com.epson.eposdevice.display;`
Keyboard class	`import com.epson.eposdevice.keyboard;`
Printer class	`import com.epson.eposdevice.printer;`
Scanner class	`import com.epson.eposdevice.scanner;`
SimpleSerial class	`import com.epson.eposdevice.simpleserial;`
CommBoxManager class	`import com.epson.eposdevice.commbox;`

7 Confirm that the target project's "/libs" folder is in the Source tab of the target project's properties. If not, add "libs" to the build path using (Add Folder...).

8 With the target project selected from Eclipse's Package Explorer, select (Preferences) in the (Window) menu.

9 The (Preferences) screen is displayed. From the list on the left, select (Java)-(Compiler).

10 The (Compiler) screen is displayed. Set the (Compiler compliance level:) to "1.6", and click (Apply). After that, click (OK).

11 Double-click (AndroidManifest.xml) from Eclipse's Package Explorer.

12 Select the (Permissions) tab.

13 The (Android Manifest Permissions) screen is displayed. Click the (Add) button.

14 Select (Uses Permission), and click the (OK) button.

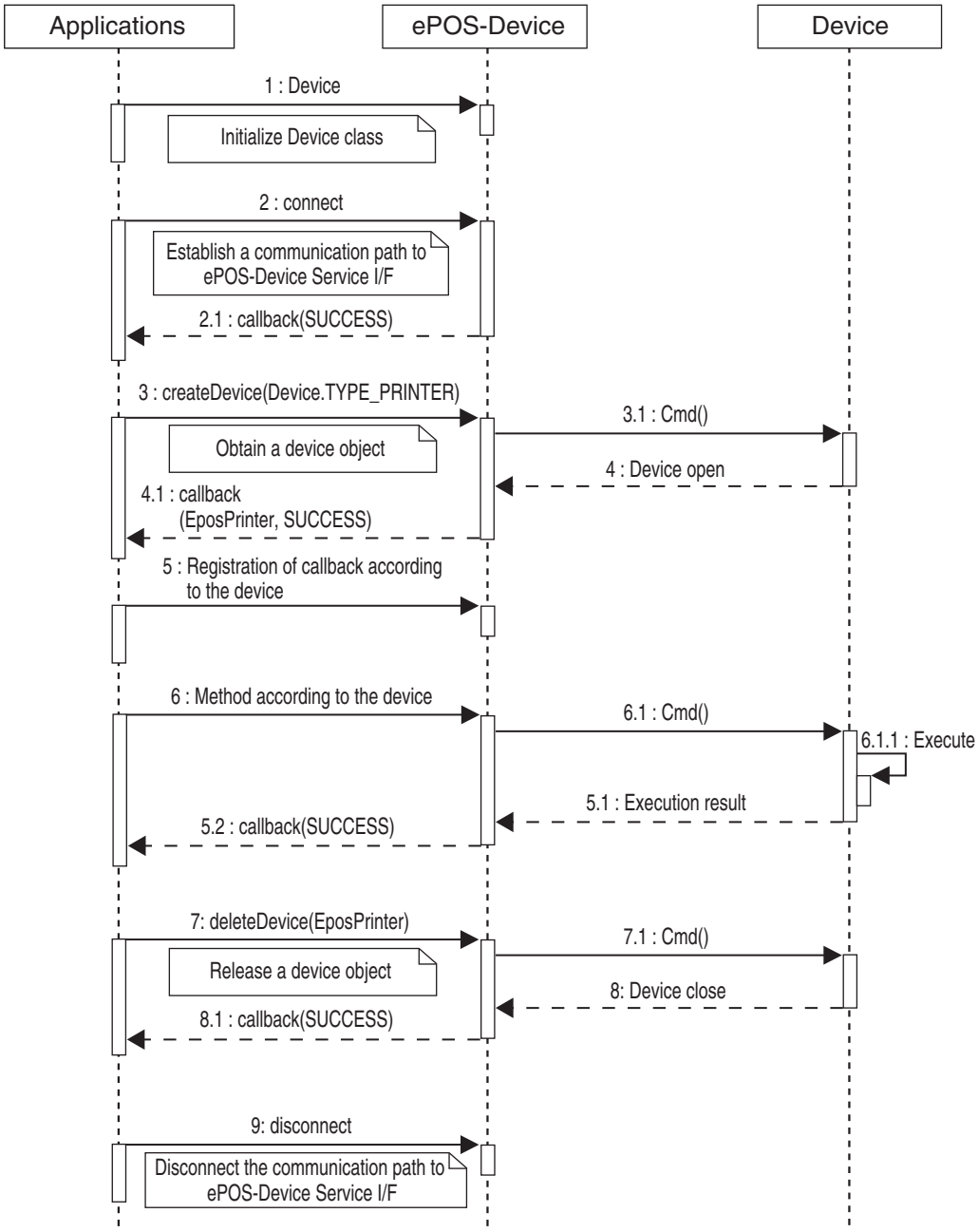
15 (Uses Permission) is added to (Permissions). Select the permissions of functionalities attached to the added (Uses Permission) from the (Name) under (Attributes for Uses Permission).

Functionality	(Name) setting
Wi-Fi	android.permission.INTERNET

16 Save "AndroidManifest.xml".

Programming Flow

The basic programming sequence of ePOS-Device SDK for Android is as follows:

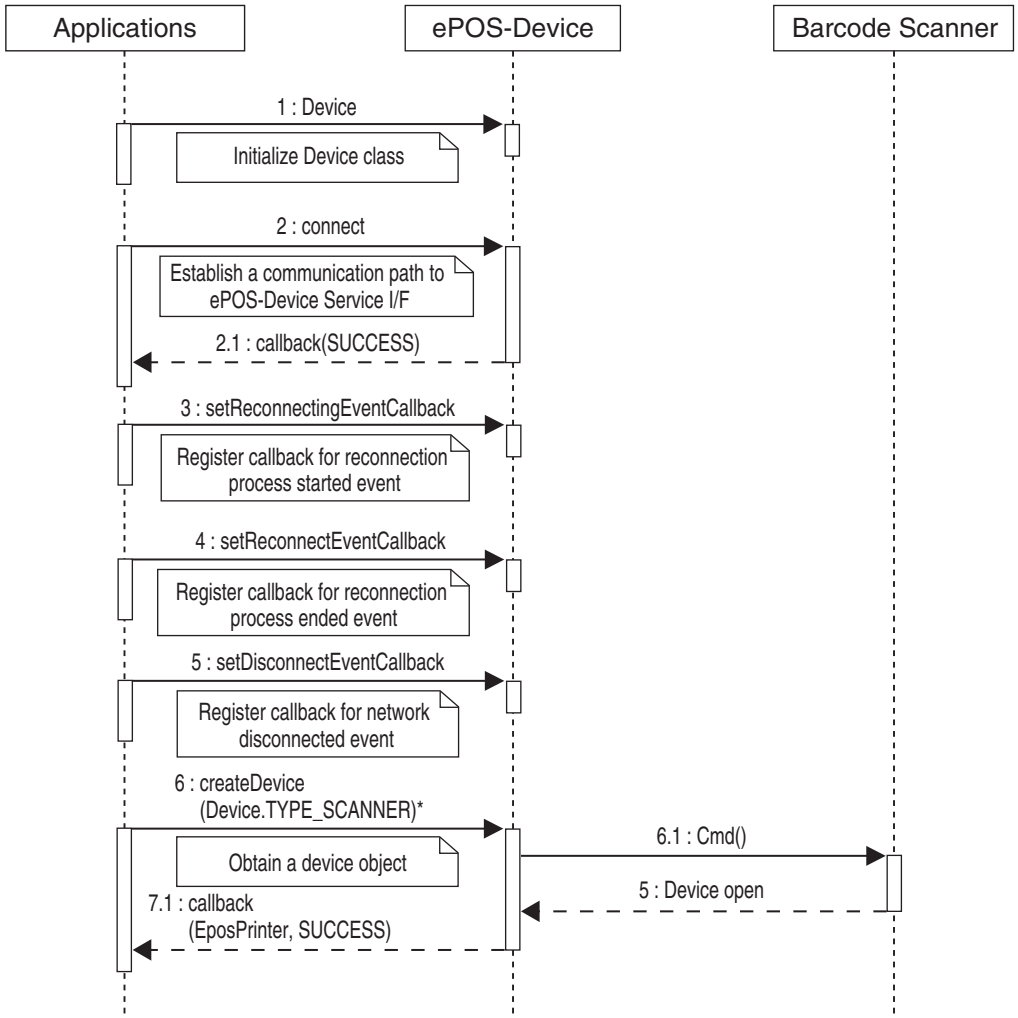


If Automatically Reconnecting to the Network

The following is a sequence for automatically reconnecting if network communication is cut off with the TM-i.

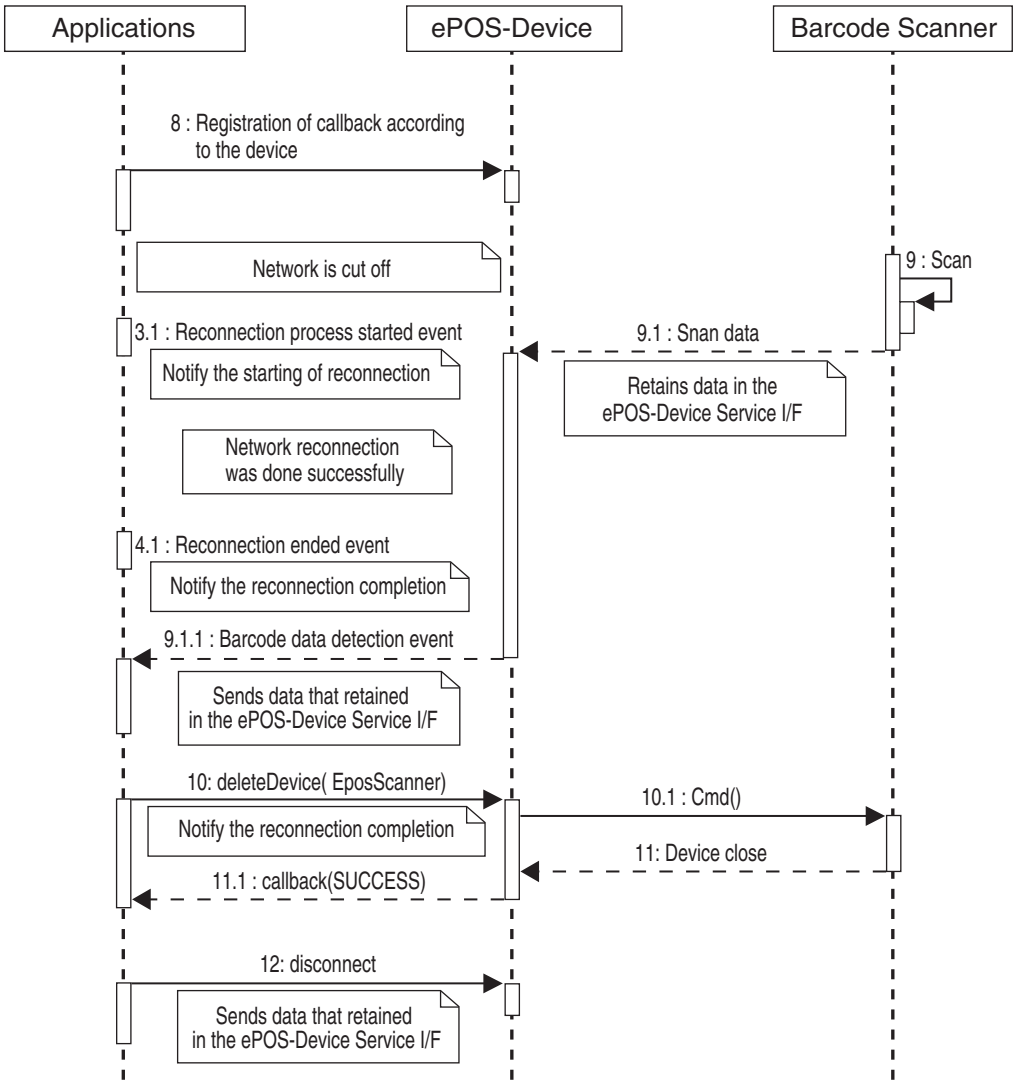
> The following uses a case where the barcode scanner is used as an example.

(1/2)



*: Sets options parameters to resend data when reconnected to the network.

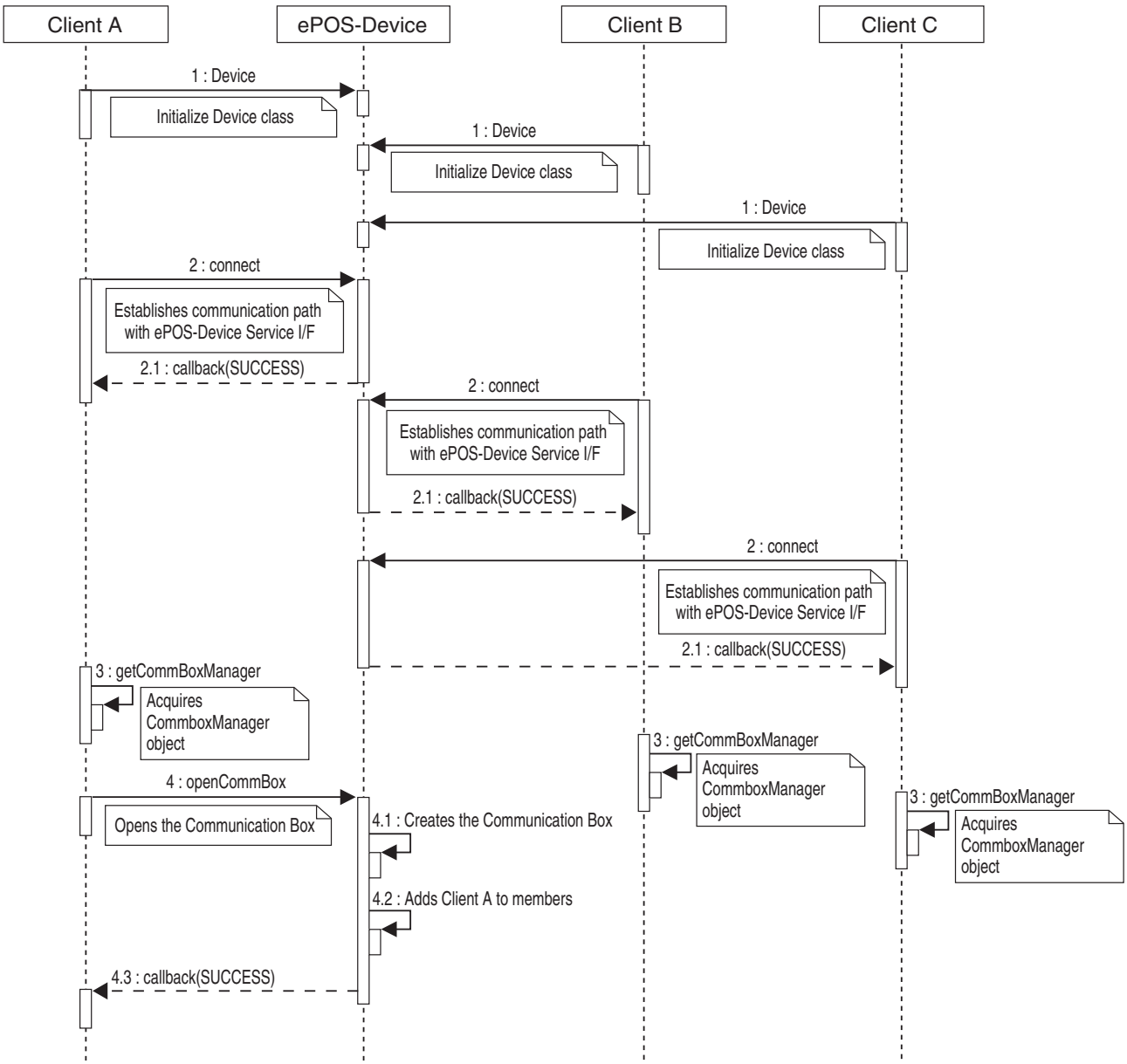
(2/2)


Applications
ePOS-Device
Barcode Scanner
8 : Registration of callback according to the device
Network is cut off
9 : Scan
3.1 : Reconnection process started event
9.1 : Snan data
Notify the starting of reconnection
Retains data in the ePOS-Device Service I/F
Network reconnection was done successfully
4.1 : Reconnection ended event
Notify the reconnection completion
9.1.1 : Barcode data detection event
Sends data that retained in the ePOS-Device Service I/F
10: deleteDevice(EposScanner)
10.1 : Cmd()
Notify the reconnection completion
11: Device close
11.1 : callback(SUCCESS)
12: disconnect
Sends data that retained in the ePOS-Device Service I/F

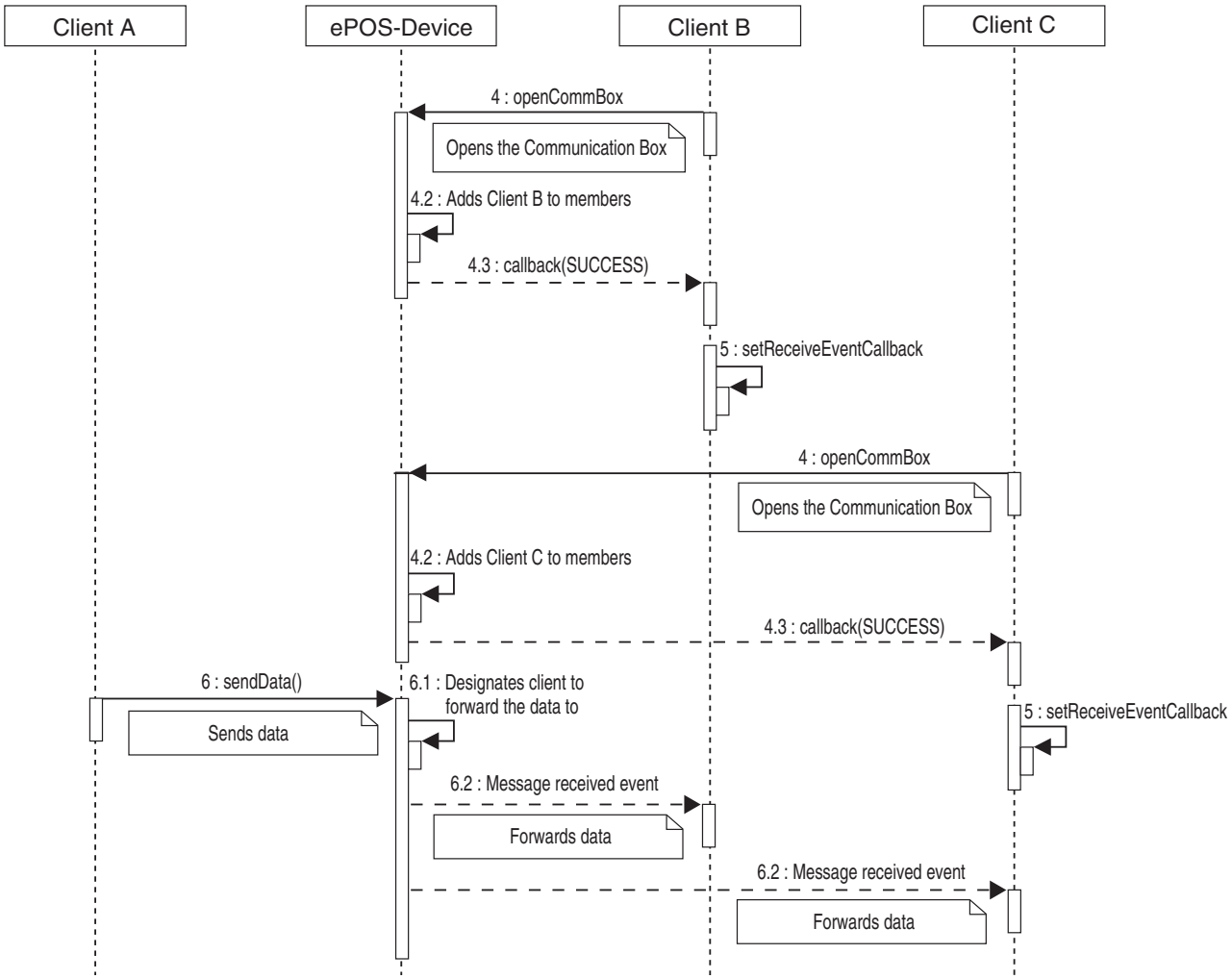
Programming flow for the Communication Box

The following indicates the basic programming sequence using the Communication Box.
A "Client" in the sequence figure refers to an application.

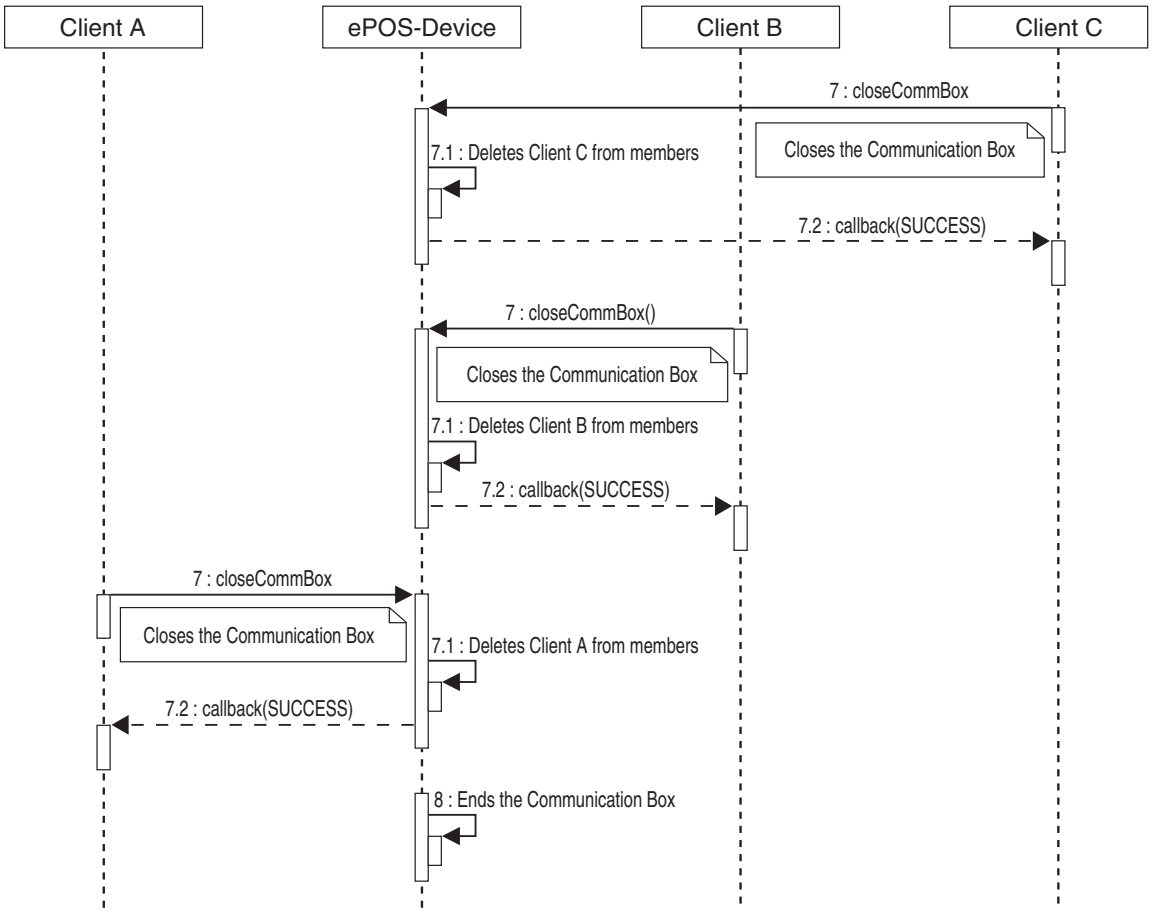
(1/3)



(2/3)


Client A
ePOS-Device
Client B
Client C
4 : openCommBox
Opens the Communication Box
4.2 : Adds Client B to members
4.3 : callback(SUCCESS)
5 : setReceiveEventCallback
4 : openCommBox
Opens the Communication Box
4.2 : Adds Client C to members
4.3 : callback(SUCCESS)
6 : sendData()
Sends data
6.1 : Designates client to forward the data to
5 : setReceiveEventCallback
6.2 : Message received event
Forwards data
6.2 : Message received event
Forwards data

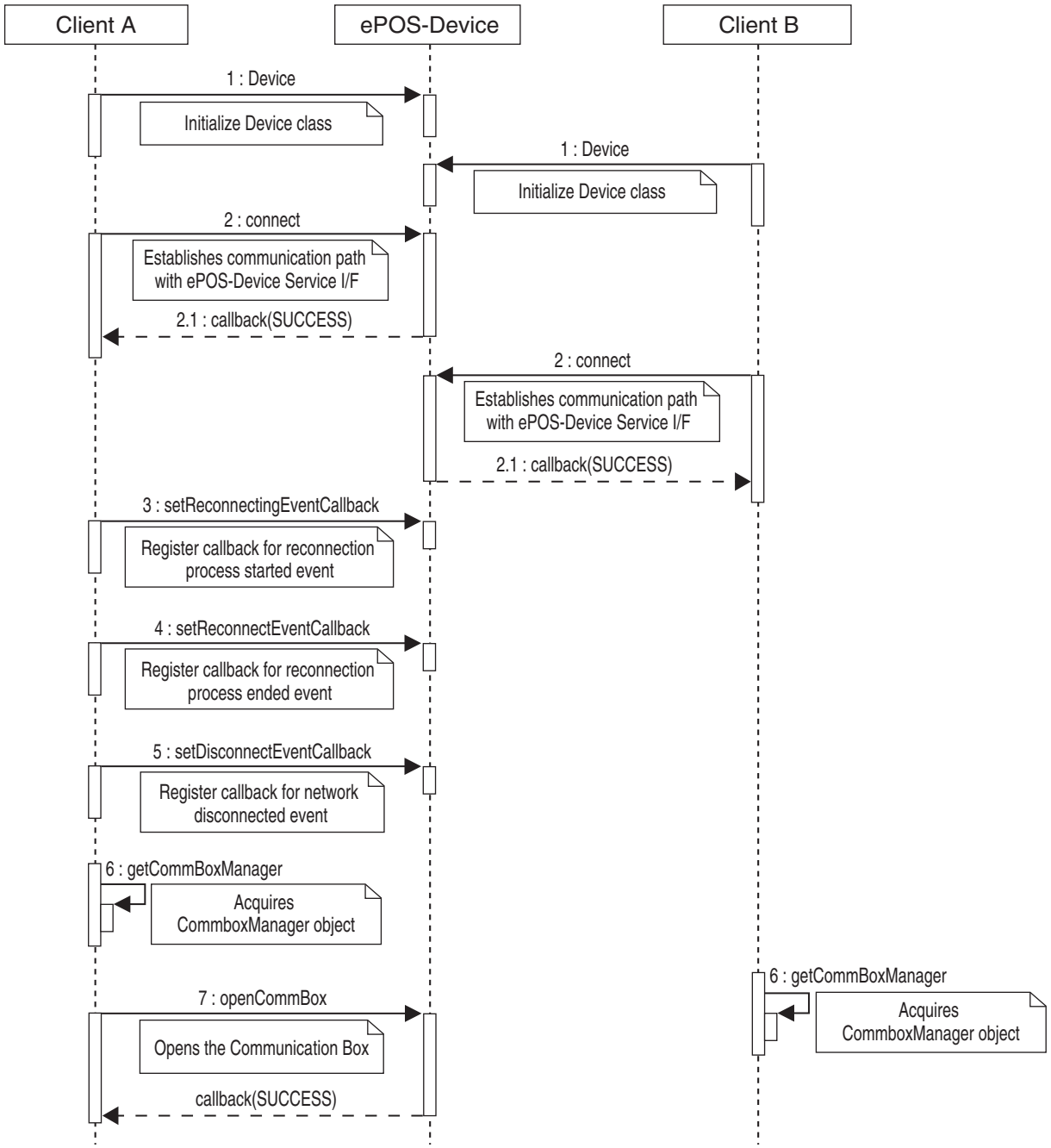
(3/3)

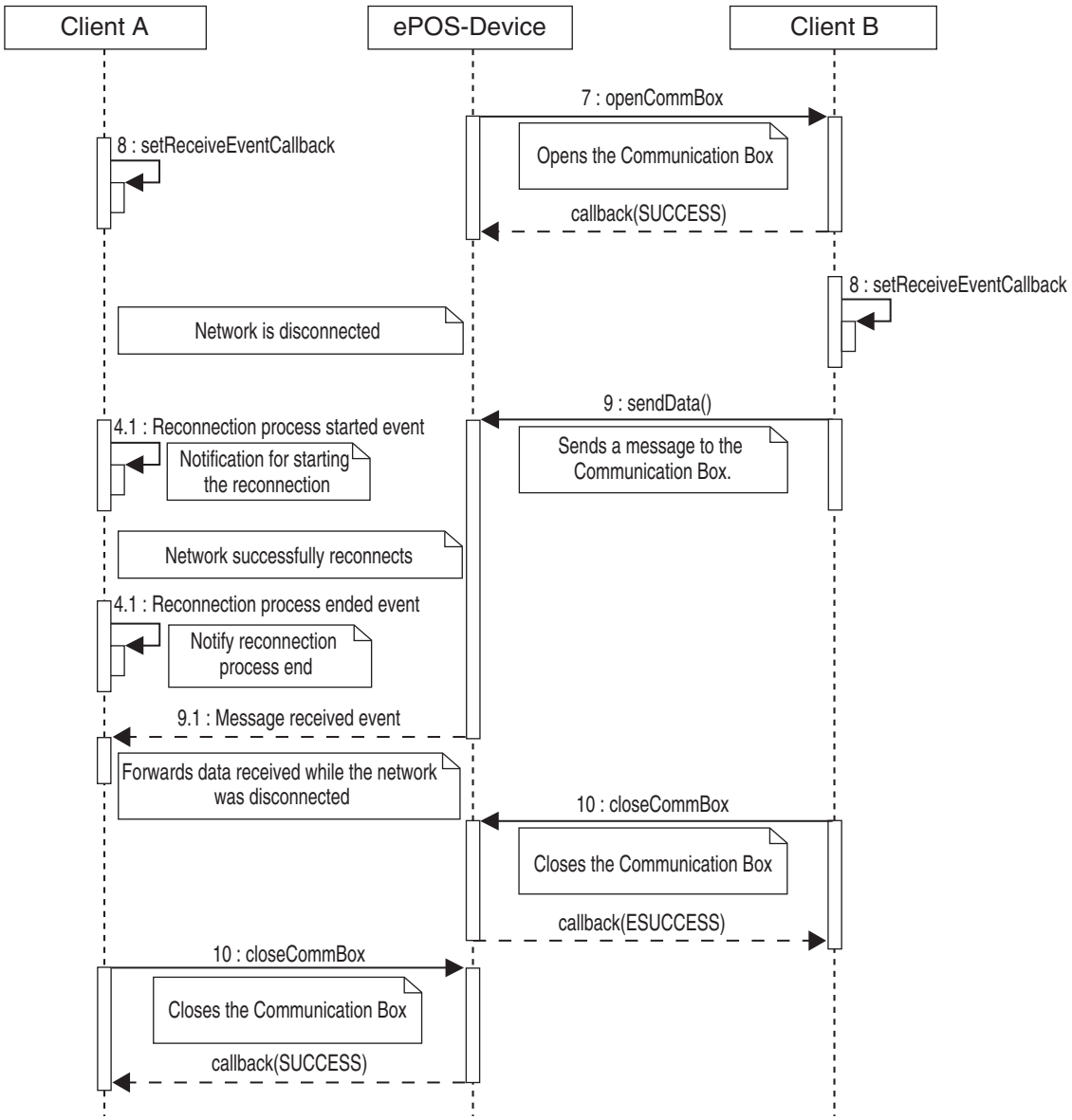

Client A
ePOS-Device
Client B
Client C
7 : closeCommBox
Closes the Communication Box
7.1 : Deletes Client C from members
7.2 : callback(SUCCESS)
7 : closeCommBox()
Closes the Communication Box
7.1 : Deletes Client B from members
7.2 : callback(SUCCESS)
7 : closeCommBox
Closes the Communication Box
7.1 : Deletes Client A from members
7.2 : callback(SUCCESS)
8 : Ends the Communication Box

If automatically reconnecting the network

If network communication with the TM-i was reconnected, the Communication Box sequence is as follows. A "Client" in the sequence figure refers to an application.

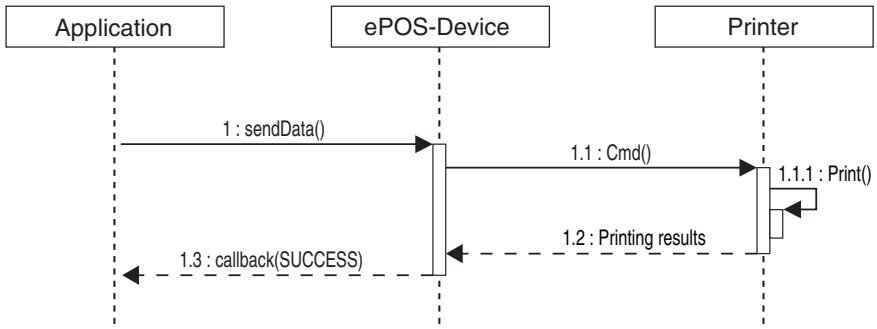
(1/2)



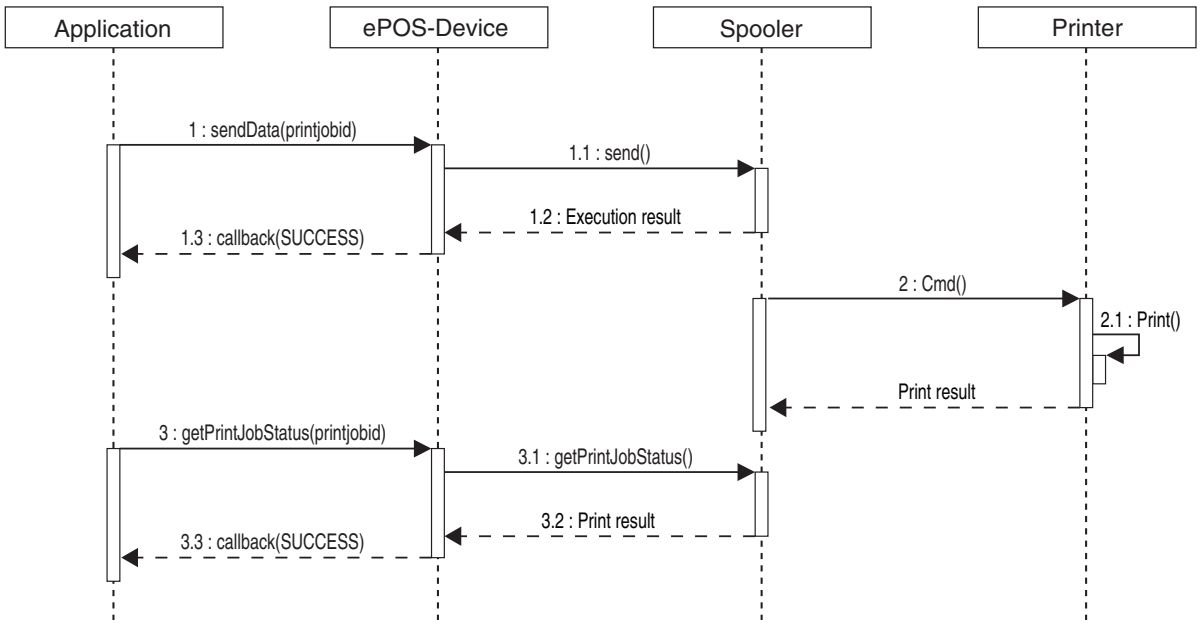

Client A
ePOS-Device
Client B
7 : openCommBox
Opens the Communication Box
8 : setReceiveEventCallback
callback(SUCCESS)
8 : setReceiveEventCallback
Network is disconnected
9 : sendData()
4.1 : Reconnection process started event
Notification for starting the reconnection
Sends a message to the Communication Box.
Network successfully reconnects
4.1 : Reconnection process ended event
Notify reconnection process end
9.1 : Message received event
Forwards data received while the network was disconnected
10 : closeCommBox
Closes the Communication Box
callback(ESUCCESS)
10 : closeCommBox
Closes the Communication Box
callback(SUCCESS)

Using the Spooler Function

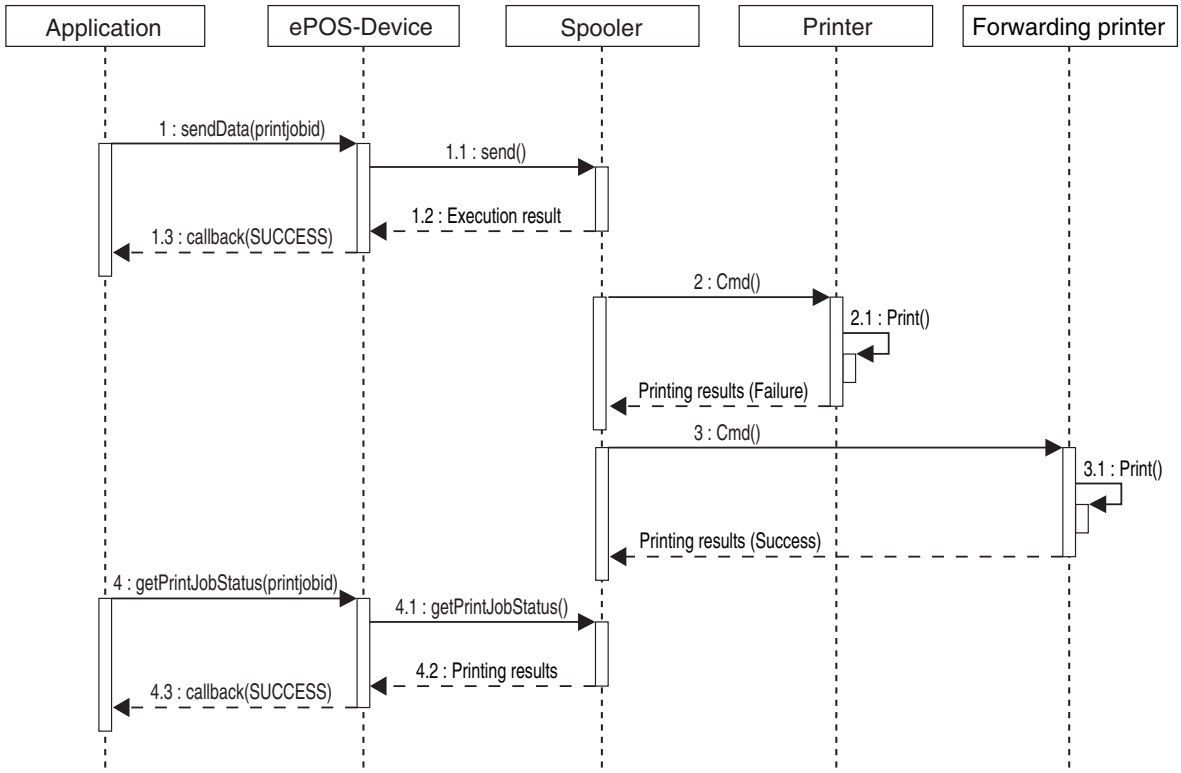
A printing job will be executed immediately and a response will be returned to the application after printing is complete when sending a printing request to ePOS-Device Service I/F from the application when the spooler function is disabled.



The print data job will be added to the queue and a response will be returned to the application without waiting for printing to complete when sending a printing request to ePOS-Device Service I/F from the application when the spooler function is enabled.



Even if the output printer cannot complete the printing task, ePOS-Device Service I/F does not return an error to the application. Enabling forward printing will allow for the printing task to be completed on a substitute printer, with the application acquiring the printing results afterward. Refer to the sequence diagram below.



Event Notification Method

The ePOS-Device SDK for Android uses a listener to notify events to application.

List of Listener Interface

For details of listener interface, refer to API Reference (p.43).

Device Object

Package: **com.epson.eposdevice.**

Function	Event listener Notification destination method Notification destination registration API
Reconnection process started event	com.epson.eposdevice.ReconnectingListener onReconnecting setReconnectingEventCallback (p.55)
Reconnection ended event	com.epson.eposdevice.ReconnectListener onReconnect setReconnectEventCallback (p.56)
Network disconnected event	com.epson.eposdevice.DisconnectListener onDisconnect setDisconnectEventCallback (p.57)

Display Object

Package: **com.epson.eposdevice.display.**

Function	Event listener Notification destination method Notification destination registration API
Control result received event	com.epson.eposdevice.display.ReceiveListener onDspReceive setReceiveEventCallback (p.70)

Keyboard Object

Package: **com.epson.eposdevice.keyboard.**

Function	Event listener Notification destination method Notification destination registration API
Key press detection event	com.epson.eposdevice.keyboard.KeyPressListener onKbdKeyPress setKeyPressEventCallback (p.72)
Character string detection event	com.epson.eposdevice.keyboard.StringListener onKbdString setStringEventCallback (p.73)

Printer Object

> The response document received event for acquiring the job ID is only compatible with TM-i firmware Ver.4.1 or later.

Package: **com.epson.eposdevice.printer.**

Function	Event listener Notification destination method Notification destination registration API
Response document received event	com.epson.eposdevice.printer.ReceiveListener onPtrReceive setReceiveEventCallback (p.124)
Response document received event (For acquiring the job ID)	com.epson.eposdevice.printer.JobReceiveListener onPtrReceive setReceiveEventCallback (p.124)
Status changed event	com.epson.eposdevice.printer.StatusChangeListener onPtrStatusChange setStatusChangeEventCallback (p.128)
Battery status changed event	com.epson.eposdevice.printer.BatteryStatusChangeListener onPtrBatteryStatusChange setBatteryStatusChangeEventCallback (p.129)
Online event	com.epson.eposdevice.printer.OnlineListener onPtrOnline setOnlineEventCallback (p.130)
Offline event	com.epson.eposdevice.printer.OfflineListener onPtrOffline setOfflineEventCallback (p.131)
Power off event	com.epson.eposdevice.printer.PowerOffListener onPtrPowerOff setPowerOffEventCallback (p.132)

Package: **com.epson.eposdevice.printer.**

Function	Event listener
	Notification destination method
	Notification destination registration API
Cover ok event	com.epson.eposdevice.printer.CoverOkListener
	onPtrCoverOk
	setCoverOkEventCallback (p.133)
Cover open event	com.epson.eposdevice.printer.CoverOpenListener
	onPtrCoverOpen
	setCoverOpenEventCallback (p.134)
Paper ok event	com.epson.eposdevice.printer.PaperOkListener
	onPtrPaperOk
	setPaperOkEventCallback (p.135)
Paper near end event	com.epson.eposdevice.printer.PaperNearEndListener
	onPtrPaperNearEnd
	setPaperNearEndEventCallback (p.136)
Pater end event	com.epson.eposdevice.printer.PaperEndListener
	onPtrPaperEnd
	setPaperEndEventCallback (p.137)
Drawer closed event	com.epson.eposdevice.printer.DrawerClosedListener
	onPtrDrawerClosed
	setDrawerClosedEventCallback (p.138)
Drawer open event	com.epson.eposdevice.printer.DrawerOpenListener
	onPtrDrawerOpen
	setDrawerOpenEventCallback (p.139)
Battery low event	com.epson.eposdevice.printer.BatteryLowListener
	onPtrBatteryLow
	setBatteryLowEventCallback (p.141)
Battery ok event	com.epson.eposdevice.printer.BatteryOkListener
	onPtrBatteryOk
	setBatteryOkEventCallback (p.140)

Scanner Object

Package: **com.epson.eposdevice.scanner.**

Function	Event listener
	Notification destination method
	Notification destination registration API
Barcode data detection event	com.epson.eposdevice.scanner.DataListener
	onScanData
	setDataEventCallback (p.142)

SimpleSerial Object

Package: **com.epson.eposdevice.simpleserial.**

Function	Event listener Notification destination method Notification destination registration API
Sent results of optional command notification event	com.epson.eposdevice.simpleserial.CommandReplyListener onSimpleSerialCommandReply setCommandReplyEventCallback (p.144)

CommboxManager Object

Package: **com.epson.eposdevice.commbox.**

Function	Event listener Notification destination method Notification destination registration API
Open event of Communication Box	com.epson.eposdevice.commbox.OpenCommBoxListener onOpenCommBox openCommBox (p.145)
Close event of Communication Box	com.epson.eposdevice.commbox.CloseCommBoxListener onCloseCommBox closeCommBox (p.147)

Commbox Object

Package: **com.epson.eposdevice.commbox.**

Function	Event listener Notification destination method Notification destination registration API
Acquired callback of sent history	com.epson.eposdevice.commbox.GetCommHistoryListener onGetCommHistory getCommHistory (p.149)
Callback of message sent	com.epson.eposdevice.commbox.SendDataListener onCommboxSendData sendData (p.151)
Callback of message received	com.epson.eposdevice.commbox.ReceiveListener onCommboxReceive setReceiveEventCallback (p.153)

API Reference

This chapter describes the APIs provided in the ePOS-Device SDK for Android.

List of ePOS-Device API

ePOS-Device SDK for Android provides the following classes:

- ❏ Device class (p. 43)
- ❏ Display class (p. 44)
- ❏ Keyboard class (p. 44)
- ❏ Printer class (p. 44)
- ❏ Scanner class (p. 46)
- ❏ SimpleSerial class (p. 47)
- ❏ CommBoxManager class (p. 47)
- ❏ CommBox class (p. 47)
- ❏ Log class (p. 47)
- ❏ EposException class (p. 47)
- ❏ EposCallbackCode class (p. 47)

When using API of each class, it is necessary to define import per class in the Java source file. For details, refer to How to Incorporate the ePOS-Device SDK for Android (p.27).

Device Class

API	Description	Page
Device	Initializes an ePOSDevice class.	48
connect	Establishes a communication path.	49
disconnect	Disconnects a communication path.	50
isConnected	Obtains communication path status.	50
createDevice	Obtains a device object.	51
deleteDevice	Deletes a device object.	53
getAdmin	Obtains the administrator information.	54
getLocation	Obtains the installation location information.	54
getCommBoxManager	Acquires CommBoxManager object	54
setReconnectingEventCallBack	Registers the reconnection started event callback method	55
setReconnectEventCallBack	Registers the reconnection completed event callback method	56
setDisconnectEventCallBack	Registers the network disconnection event callback method	57

Display Class

API	Description	Page
sendData	Sends a control instruction.	58
ClearCommandBuffer	Clears the command buffers.	58
reset	Reset	59
createWindow	Define the display area.	60
destroyWindow	Destroys the display area.	61
setCurrentWindow	Switches the display area.	61
clearWindow	Clears the current display area.	62
setCursorPosition	Moves the cursor position.	62
moveCursorPosition	Moves the cursor position in the current display area.	63
setCursorType	Changes the cursor display.	64
addText	Displays text.	65
addReverseText	Displays reversed text.	66
addMarquee	Displays marquee.	67
setBlink	Display blinking.	68
setBrightness	Changes the display brightness.	68
showClock	Displays the clock.	69
addCommand	Executes a command.	69
setReceiveEventCallback	Registers the control result receive event callback method.	70

Keyboard Class

API	Description	Page
setPrefix	Specifies the conditions for handling input as a series of strings.	71
setKeypressEventCallback	Registers the key press detection event callback method.	72
setStringEventCallback	Registers the string detection event callback method.	73

Printer Class

API	Description	Page
sendData	• Sends a print document. • Specifies the job ID and sends the print document. *	74
getPrintJobStatus *	Acquires print job status.	75
clearCommandBuffer	Clears the command buffers.	75
addTextAlign	Adds the text alignment settings to the command buffer.	76
addTextLineSpace	Adds the line feed space settings to the command buffer.	76
addTextRotate	Adds the text rotation print settings to the command buffer.	77
addText	Adds text printing to the command buffer.	78

API	Description	Page
addTextLang	Adds the target language settings to the command buffer.	79
addTextFont	Adds the text font settings to the command buffer.	80
addTextSmooth	Adds the text smoothing settings to the command buffer.	81
addTextDouble	Adds the double-sized text settings to the command buffer.	82
addTextSize	Adds the text scale settings to the command buffer.	83
addTextStyle	Adds the text style settings to the command buffer.	84
addTextPosition	Adds the print position settings to the command buffer.	86
addTextVPosition	Adds a tag for specifying the print vertical position of text.	86
addFeedUnit	Adds a tag for paper feeding (in dots).	87
addFeedLine	Adds a tag for paper feeding (in lines).	87
addFeedPosition	Adds control of label paper/black mark paper to command buffer.	88
addFeed	Adds a line feed to the command buffer.	88
addImage	Adds raster image printing to the command buffer.	89
addLogo	Adds NV logo printing to the command buffer.	91
addBarcode	Adds barcode printing to the command buffer.	92
addSymbol	Adds 2D-Code printing to the command buffer.	96
addHLine	Adds horizontal line printing to the command buffer.	101
addVLineBegin	Adds vertical line start to the command buffer.	102
addVLineEnd	Adds vertical line end to the command buffer.	103
addPageBegin	Adds page mode start to the command buffer.	104
addPageEnd	Adds page mode end to the command buffer.	104
addPageArea	Adds the print area settings in the page mode to the command buffer.	105
addPageDirection	Adds the print direction settings in the page mode to the command buffer.	106
addPagePosition	Adds the print position settings in the page mode to the command buffer.	107
addPageLine	Adds drawing a line in the page mode to the command buffer.	108
addPageRectangle	Adds drawing a rectangle in the page mode to the command buffer.	109
addCut	Adds paper cut to the command buffer.	110
addPulse	Adds drawer kick-out to the command buffer.	111
addSound	Adds turning on the buzzer to the command buffer.	112
addLayout	Adds the paper layout setup to command buffer	114
addCommand	Adds a command to the command buffer.	117
addRecovery	Adds a tag for recovering from an error.	117
addReset	Adds a tag for resetting the printer.	118

API	Description	Page
startMonitor	Enables a status event.	118
stopMonitor	Disables a status event.	118
recover	Recovers from an error that can be recovered from.	119
reset	Resets the printer.	119
halftone property	Raster image halftone processing method.	120
brightness property	Raster image brightness correction value.	121
force property	Forced transmission mode.	122
timeout property	Sending timeout time.	123
interval property	Printer status update interval.	123
setReceiveEventCallback	• Registers the receive response document event callback method. • Registers the job ID response document received event callback method *	124
setStatuschangeEventCallback	Registers the status change event.	128
setBatteryStatusChangeEventCallback	Registers the battery status change event callback method.	129
setOnlineEventCallback	Registers the online event callback method.	130
setOfflineEventCallback	Registers the offline event callback method.	131
setPowerOffEventCallback	Registers the non-response event callback method.	132
setCoverOkEventCallback	Registers the cover close event callback method.	133
setCoverOpenEventCallback	Registers the cover open event callback method.	134
setPaperOkEventCallback	Registers the paper remaining event callback method.	135
setPaperNearEndEventCallback	Registers the paper near end event callback method.	136
setPaperEndEventCallback	Registers the paper end event callback method.	137
setDrawerClosedEventCallback	Registers the drawer close event callback method.	138
setDrawerOpenEventCallback	Registers the drawer open event callback method.	139
setBatteryOkEventCallback	Registers the battery ok event callback method.	140
setBatteryLowEventCallback	Registers the battery low event callback method.	141

* TM-i firmware Ver.4.1 or later versions supported

Scanner Class

API	Description	Page
setDataEventCallback	Registers the barcode data detection event callback method.	142

SimpleSerial Class

API	Description	Page
sendCommand	Sends a command.	143
setCommandreplyEventCallback	Registers the command sending result notification event callback method.	144

CommBoxManager Class

API	Description	Page
openCommBox	Opens the Communication Box.	145
closeCommBox	Closes the Communication Box.	147

CommBox Class

API	Description	Page
getCommHistory	Acquires the message transfer history.	149
sendData	Sends a message to the Communication Box.	151
setReceiveEventCallback	Registers the notifies that the message in the Communication Box was received event callback method.	153

Log Class

API	Description	Page
setLogSettings	Sets the log output function.	154

EposException Class

API	Description	Page
getErrorStatus	Acquires error status when an exception is thrown.	156

EposCallbackCode Class

Class that has a constant of error status passed to the notification destination method of callback.

Device Class

Device

Initialize a Device class instance.

Syntax

```
public Device(Context context) throws EposException
```

Parameter

- context: Specifies the context of the application.
  This parameter is not designated if the log output function is not used.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_MEMORY	Could not allocate memory.

connect

Establishes a communication path to ePOS-Device Service I/F of TM-i.

Syntax

```
public void connect(String ipAddress,
          ConnectListener  target) throws EposException
```

Parameter

- ipAddress: Specifies the IP address of TM-i.
- target : This specifies the object (listener interface) that has the notification destination method.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_ILLEGAL	Attempted to restart communication for a device that already started communication.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.
EposException.ERR_PROCESSING	Could not execute a process.

Listener Interface

```
public interface ConnectListener extends EventListener
```

Listener Registration Method

```
void onConnect (String ipAddress, int code)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- code: Sets error status of processing results.

Error status	Description
EposCallbackCode.SUCCESS	Connected successfully.
EposCallbackCode.ERR_TIMEOUT	A timeout occurred.

disconnect

Disconnects the communication path connected with "connect".

Syntax

```
public void disconnect() throws EposException
```

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_ILLEGAL	This API was called when communication hasn't been started.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.
EposException.ERR_PROCESSING	Could not execute a process.

isConnected

Obtains status of the communication path connected with connect (p.49).

If the terminal goes to sleep, the returned value may not be the correct value.

Syntax

```
public boolean isConnected()
```

Return value

The connection status of the communication path

Return value	Description
true	Connected
false	Not connected

createDevice

Obtains a device object to act as the interface with the device.

- When this API is successfully executed, the specified device is exclusively locked and calling "createDevice" from a different browser returns "EposException.ERR_IN_USE". The printer is not locked exclusively and can obtain device objects with "createDevice" from multiple browsers.
- If "createDevice" is executed immediately during a return process after an ondisconnect event, "EposException.ERR_IN_USE" occurs. In this case, retry until a value other than "EposException.ERR_IN_USE" is obtained.

Syntax

```
public void createDevice
            (String deviceId, int deviceType,
             int crypto int buffer,
             CreateDeviceListener target)
             throws EposException
```

Parameter

- deviceId: Specifies the device ID.

  - When specifying "Device.TYPE_DISPLAY" to the second parameter (deviceType), specify "local_display".
  - When specifying "Device.TYPE_SIMPLE_SERIAL" to the second parameter (deviceType), specify "local_serial".

- deviceType: Specifies any of the following device types:

Setting	Description
Device.TYPE_PRINTER	Specifies the printer for the device type.
Device.TYPE_DISPLAY	Specifies the customer display for the device type.
Device.TYPE_KEYBOARD	Specifies the keyboard for the device type.
Device.TYPE_SCANNER	Specifies the scanner for the device type.
Device.TYPE_SIMPLE_SERIAL	Specifies the serial communication device for the device type.

- crypto: Specifies whether encryption of the communication data is required.

Setting	Description
Device.FALSE	Disable

- buffer: Specifies if it is necessary to buffer the data that occurred in the device during reconnection and resend the data upon reconnection.

Setting	Description
Device.TRUE	Enable
Device.FALSE	Disable

- target : This specifies the object (listener interface) that has the notification destination method.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_CONNECT	Could not communicate.
EposException.ERR_IN_USE	Device in use exists.
EposException.ERR_FAILURE	An unspecified error occurred.
EposException.ERR_PROCESSING	Could not execute a process.

Listener Interface

```
public interface CreateDeviceListener
                 extends EventListener
```

Listener Registration Method

```
void onCreateDevice (String ipAddress, String deviceId,
                     int deviceType,
                     Object deviceObject, int code)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.
- deviceType: Sets a device deviceType specified in this API.
- deviceObject:
  Sets a device object corresponding to deviceType specified in this API.

deviceType	Device object
Device.TYPE_PRINTER	EposPrinter
Device.TYPE_DISPLAY	EposDisplay
Device.TYPE_KEYBOARD	EposKeyboard
Device.TYPE_SCANNER	EposScanner
Device.TYPE_SIMPLE_SERIAL	EposSimpleSerial

- code: Sets error status of processing results.

Error status	Description
EposCallbackCode.SUCCESS	Obtained the device object successfully.
EposException.ERR_CONNECT	Could not communicate.
EposCallbackCode.ERR_NOT_FOUND	Device is not found.
EposCallbackCode.ERR_IN_USE	Device is in use.
EposCallbackCode.ERR_OPEN	Failed to open the device.
EposCallbackCode.ERR_TYPE_INVALID	Invalid device type.
EposCallbackCode.ERR_FAILURE	Other error occurred.

deleteDevice

Releases a device to be available from other clients.

Syntax

```
public void deleteDevice(Object deviceObject,
                         DeleteDeviceListener target)
                         throws EposException
```

Parameter

- deviceObject:Specifies the device object.
- target : This specifies the object (listener interface) that has the notification destination method.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_CONNECT	Could not communicate.
EposException.ERR_FAILURE	An unspecified error occurred.
EposException.ERR_PROCESSING	Could not execute a process.

Listener Interface

```
public interface DeleteDeviceListener
                 extends EventListener
```

Listener Registration Method

```
void onDeleteDevice (String ipAddress, String deviceId,
                     int code)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.
- code: Sets error status of processing results.

Error status	Description
EposCallbackCode.SUCCESS	Device close succeeded.
EposException.ERR_CONNECT	Could not communicate.
EposCallbackCode.ERR_NOT_FOUND	Device not open.
EposCallbackCode.ERR_CLOSE	Failed to close the device.
EposCallbackCode.ERR_FAILURE	Other error occurred.

getAdmin

Obtains the administrator name set in TM-i.

The string of the administrator name can be set with EPSON TMNet WebConfig.

Syntax

```
public String getAdmin()
```

Return value

Returns character string of administrator name.

getLocation

Obtains the installation location information set in TM-i.

The string of the installation location can be set with EPSON TMNet WebConfig.

Syntax

```
public String getLocation()
```

Return value

Returns the installation location information.

getCommBoxManager

Obtains the CommBoxManager object that performs communication between applications.

Syntax

```
public CommBoxManager getCommBoxManager()
```

Return value

Returns the Constructor of class CommBoxManager.

setReconnectingEventCallback

This registers a callback method that receives the information that a reconnection process has started.

When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setReconnectingEventCallback
                            (ReconnectingListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface ReconnectingListener
                    extends EventListener
```

Listener Registration Method

```
void onReconnecting(String ipAddress)
```

Parameter

- ipAddress: Sets IP address of TM-i.

setReconnectEventCallback

This registers a callback method that receives the information that a reconnection process has ended.

When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setReconnectEventCallback
                              (ReconnectListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface ReconnectListener extends EventListener
```

Listener Registration Method

```
void onReconnect(String ipAddress)
```

Parameter

- ipAddress: Sets IP address of TM-i.

setDisconnectEventCallback

This registers a callback method that receives the information that a reconnection process has failed.

If this event occurred, perform processing for taking measures against network disconnections, such as establishing a side connection.

When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setDisconnectEventCallback
                        (DisconnectListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface DisconnectListener
                 extends EventListener
```

Listener Registration Method

```
void onDisconnect(String ipAddress)
```

Parameter

- ipAddress: Sets IP address of TM-i.

Display Class

sendData

Sends commands in the command buffer.

Syntax

```
public void sendData() throw EposException
```

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_CONNECT	Could not communicate.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.
EposException.ERR_PROCESSING	Could not execute a process.

clearCommandBuffer

Clears command buffers.

Syntax

```
public void clearCommandBuffer();
```

reset

Adds the initialization settings of the customer display to the command buffer. Initialization changes the status as follows:

- All the displayed characters are cleared and all the registered windows are deleted.
- The cursor is restored to the default status and moves to the customer display origin.
- The display blinking and brightness settings are restored to the default status.

Syntax

```
public void reset() throws EposException
```

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

createWindow

Adds the window settings to the command buffer.

> ⚠ Define a window so that it does not overlap with a defined window.
> Specify the size to fit in the customer display size (20 digits x 2 rows).

Syntax

```
public void createWindow(int number, int x, int y,
                         int width, int height, int scrallMode)
                         throw EposException
```

Parameter

- number: Specifies the window number. Specify a number between 1 and 4.
- x: Specifies the x-coordinate of the window origin on the customer display. Specify a number between 1 and 20.
- y: Specifies the y-coordinate of the window origin on the customer display. Specify "1" or "2".
- width: Specifies the window width.
- height: Specifies the window height.
- scrollMode: Specifies the window scroll mode from the following:

Setting	Description
Display.SCROLL_OVERWRITE	When characters are displayed while the display position is at the upper right end, the display position moves to the lower left end. When characters are displayed while the display position is at the lower right end, the position moves to the upper left end.
Display.SCROLL_VERTICAL	When characters are displayed while the display position is at the upper right end, the display position moves to the lower left end. When characters are displayed while the display position is at the lower right end, the characters already displayed in the lower row are scrolled to the upper row and the lower row is cleared.
Display.SCROLL_HORIZONTAL	When characters are displayed while the display position is at the right end, all the characters already displayed in the row where the cursor exists are scrolled one character to the left and a new character is displayed at the right end.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

destroyWindow

Adds the destroy settings for the defined window to the command buffer.

Syntax

```
public void destroyWindow(int number)
                          throw EposException
```

Parameter

- number: Specifies the window number to be destroyed. Specify a number between 1 and 4.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

setCurrentWindow

Adds the change settings of the current window to the command buffer.
Sets the specified window as the current window and moves the cursor to the origin of the current window.

Syntax

```
public void setCurrentWindow(int number)
                             throw EposException
```

Parameter

- number: Specifies the destination window number. Specify a number between 1 and 4.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

clearWindow

Adds the current window display clear to the command buffer.

Syntax

```
public void clearWindow() throw EposException
```

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

setCursorPosition

Adds the cursor position settings to the command buffer.

Syntax

```
public void setCursorPosition(int x, int y)
                              throw EposException
```

Parameter

- x: Specifies the x-coordinate of the destination. Specify a number between 1 and 20.
- y: Specifies the y-coordinate of the destination. Specify "1" or "2".

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

moveCursorPosition

Adds the cursor position settings in the current window to the command buffer.

Syntax

```
public void moveCurrentPosition
                    (int position) throw EposException
```

Parameter

- position: Specifies the cursor destination in the current window by selecting from the following:

Setting	Description
Display.MOVE_TOP_LEFT	Sets the destination at the top left.
Display.MOVE_TOP_RIGHT	Sets the destination at the top right.
Display.MOVE_BOTTOM_LEFT	Sets the destination at the bottom left.
Display.MOVE_BOTTOM_RIGHT	Sets the destination at the bottom right.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

setCursorType

Adds the cursor display settings to the command buffer.

Syntax

```
public void setCursorType(int type) throw EposException
```

Parameter

- type: Specifies the cursor display method by selecting from the following:

Setting	Description
Display.CURSOR_NONE	No cursor display
Display.CURSOR_UNDERLINE	Underline

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addText

Adds string display to the command buffer.

Syntax

❏ `public void addText(String data) throws EposException`

❏ `public void addText(String data, int lang)`
`throws EposException`

❏ `public void addText(String data, int x, int y)`
`throws EposException`

❏ `public void addText(String data, int x, int y,`
`int lang) throws EposException`

Parameter

- data: Specifies the string to be displayed.
- lang: Specifies the language to display the string from the following:

Setting	Description
Display.LANG_EN(Default)	Specifies English for the display language.
Display.LANG_JA	Specifies Japanese for the display language.

- x: Specifies the x-coordinate of the display position. Specify a number between 1 and 20.
- y: Specifies the y-coordinate of the display position. Specify "1" or "2".

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addReverseText

Adds reversed string display to the command buffer.

Syntax

❏ `public void addReverseText(String data)`
`throws EposException`

❏ `public void addReverseText(String data, int lang)`
`throws EposException`

❏ `public void addReverseText(String data, int x, int y)`
`throws EposException`

❏ `public void addReverseText(String data, int x, int y,`
`int lang) throws EposException`

Parameter

- data: Specifies the string to be displayed.
- lang: Specifies the language to display the string from the following:

Setting	Description
Display.LANG_EN(Default)	Specifies English for the display language.
Display.LANG_JA	Specifies Japanese for the display language.

- x: Specifies the x-coordinate of the display position. Specify a number between 1 and 20.
- y: Specifies the y-coordinate of the display position. Specify "1" or "2".

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addMarquee

Adds marquee display to the command buffer.

Marquee is displayed in the horizontal scroll mode in one line, regardless of the scroll mode set for the display window.

Syntax

```
public void addMarquee(String data ,int format,
                       int unitWait, int repeatWait,
                       int repeatCount, int lang)
                       throws EposException
```

Parameter

- data: Specifies the string to be displayed.
- format: Specifies the string display method by selecting from the following:

Setting	Description
Display.MARQUEE_WALK	Displays the string from the right end of the window.
Display.MARQUEE_PLACE	Displays the string from the left end of the window.

- unitWait: Specifies the display interval (ms) for each character.
  A value between 0 and 2,000 can be specified.
- repeatWait: Specifies the interval (ms) to repeat the string display.
  A value between 100 and 2,000 can be specified.
- repeatCount: Specifies the number of times to repeat the string display.
  A value between 0 and 127 can be specified.
  When "0" is specified, display is repeated unlimitedly.
- lang: Specifies the language to display the string from the following:

Setting	Description
Display.LANG_EN(Default)	Specifies English for the display language.
Display.LANG_JA	Specifies Japanese for the display language.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

setBlink

Adds the screen blinking settings to the command buffer.

Syntax

```
public void setBlink(int interval) throws EposException
```

Parameter

- interval: Specifies the blinking time (ms). Specify a number between 0 and 12700.
  The specified values are rounded in units of 50ms. When "0" is specified, the display lights on.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

setBrightness

Adds the customer display brightness settings to the command buffer.

Syntax

```
publice void setBrightness(int brightness)
                           throws EposException;
```

Parameter

- brightness: Specifies the customer display brightness level by selecting from the following:

Setting	Description
Display.BRIGHTNESS_20	Set the customer display brightness to "20%".
Display.BRIGHTNESS_40	Set the customer display brightness to "40%".
Display.BRIGHTNESS_60	Set the customer display brightness to "60%".
Display.BRIGHTNESS_100	Set the customer display brightness to "100%".

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

showClock

Adds time display to the command buffer.

Enabling time display clears all the displayed strings. Sending another command during time display finishes time display.

Syntax

```
public void showClock() throws EposException
```

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addCommand

Adds commands to the command buffer.

- For information about ESC/POS commands, see the ESC/POS Command Reference.
  You can view the ESC/POS Command Reference from https://reference.epson-biz.com/.

Syntax

```
public void addCommand(byte[] data) throws EposException
```

Parameter

- data: Specifies ESC/POS command with binary data.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

setReceiveEventCallback

This registers a callback method that receives the information that the response document received event.

When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setReceiveEventCallback
                        (ReceiveListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface ReceiveListener extends EventListener
```

Listener Registration Method

```
void onDspReceive(String ipAddress, String deviceId,
                  int success,int code)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.
- success: Sets execution results of display.

Error status	Description
Display.TRUE	Process succeeded.
Display.FALSE	Process failed.

- code: Sets execution results of process.

Error status	Description
EposCallbackCode.SUCCESS	Processing was successful.
EposCallbackCode.ERR_NOT_FOUND	Device is not found.
EposCallbackCode.ERR_OPEN	Failed to open the device.
EposCallbackCode.ERR_INVALID_WINDOW	The specified window is not registered.
EposCallbackCode.ERR_PORT	An internal communication error with the device occurred.
EposCallbackCode.ERR_TIMEOUT	A timeout error occurred in communication with the device.
EposCallbackCode.ERR_PARAM	An invalid parameter is detected.

Keyboard Class

setPrefix

Sets the keycode to identify the beginning of a string when receiving keyboard input collectively as a series of strings.

Settings are cancelled if NULL or " " (empty space) is input.

Syntax

```
public void setPrefix(byte[] data) throws EposException
```

Parameter

- data: Specifies the array of keycodes. For the keycodes, refer to List of KeyCode (p.165).

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_CONNECT	Failed to communicate with a device.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

setKeyPressEventCallback

Registers a callback method that receives the key pushed event.

When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setKeyPressEventCallback
                        (KeyPressListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface KeyPressListener extends EventListener
```

Listener Registration Method

```
void onKbdKeyPress(String ipAddress, String deviceId,
                   int keyCode, String ascii)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.
- keycode: Sets a keycode pressed.
- ascii: Sets a character corresponding to a keycode.
  Sets "" (blank character) when there is no character corresponding to a keycode.

setStringEventCallback

This registers a callback method that receives character input as character string detected before Enter key is detected and starting with any of the characters specified in setPrefix (p.71).

When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setStringEventCallback
                          (StringListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface StringListener extends EventListener
```

Listener Registration Method

```
void onKbdString(String ipAddress, String deviceId,
                 String input, int prefix)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.
- input: Sets detected character string.
- prefix: Sets a keycode determined as the beginning of character string.

Printer Class

sendData

Sends an command buffer. The job ID can be specified before sending the command buffer.

- The results of executing this API will be returned to a setReceiveEventCallback (p.124).
- Execution of this API results in the value of the force property (p.122) as Printer.FALSE.

Syntax

❏ `public void sendData() throws EposException`

❏ For acquiring the job ID

```
public void sendData(String printJobId)
                              throws EposException
```

The syntax used to specify job IDs is only compatible with TM-i firmware Ver.4.1 or later. For details, refer to Specifying the Print Job ID from the Application (p.16).

Parameter

- printJobId : Specify the print job ID. (TM-i firmware Ver.4.1 or later versions)
  1 to 30 alphanumeric characters, including underscores, hyphens and periods, may be used.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_CONNECT	Failed to communicate with a device.
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.
EposException.ERR_PROCESSING	Could not execute a process.

getPrintJobStatus

Acquires print job status. (TM-i firmware Ver.4.1 or later versions)

- The results of executing this API will be returned to a setReceiveEventCallback (p.124).
- The latest print job status will be obtained when executing multiple printing processes with the same job ID.

Syntax

```
public void getPrintJobStatus(String printJobId)
                              throws EposException
```

Parameter

- printJobId : Specifies the print job ID to be verified.
  1 to 30 alphanumeric characters, including underscores, hyphens and periods, may be used.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_CONNECT	Failed to communicate with a device.
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.
EposException.ERR_PROCESSING	Could not execute a process.

clearCommandBuffer

Clears command buffers.

Syntax

```
public void clearCommandBuffer();
```

addTextAlign

Adds the text alignment setting to the command buffer.

- This API setting is applied to raster image/NV logo/barcode/2D-Code.
- In the page mode, this API specification cannot be used.
  In the page mode, use the addTextPosition (p.86) to designate the horizontal print position.

Syntax

```
public void addTextAlign(int align) throws EposException
```

Parameter

- align : Specifies the text alignment.

Setting	Description
Printer.ALIGN_LEFT (default)	Alignment to the left
Printer.ALIGN_CENTER	Alignment to the center
Printer.ALIGN_RIGHT	Alignment to the right

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addTextLineSpace

Adds the line feed space setting to the command buffer.

Syntax

```
public void addTextLineSpace(int linespc)
                             throws EposException
```

Parameter

- linespc : Specifies the line feed space (in dots). Specifies an integer from 0 to 255.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addTextRotate

Adds the text rotation setting to the command buffer.

- This API setting also applies to barcodes/2D-Codes.
- When the page mode is selected for the print mode, to set text rotation, use the addPageDirection (p.106) instead of this API function.

Syntax

```
public void addTextRotate(int rotate)
                          throws EposException
```

Parameter

- rotate : Specifies whether to rotate text.

Setting	Description
Printer.TRUE	Specifies rotated printing of text.
Printer.FALSE (default)	Cancels rotated printing of text.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addText

Adds the printing of text to the command buffer.

	After printing text, to print content other than text, execute line feed or paper feed.
	In page mode, characters are laid out in the current print position with the reference point being the character baseline dot (Printer Specifications (p.166)).

Syntax

```
public void addText(String data) throws EposException
```

Parameter

- data : Specify a character string to be printed.
  For the horizontal tab/line feed, use the following escape sequences:

String	Description
\t	Horizontal tab(HT)
\n	Line feed (LF)
\\	Carriage return

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addTextLang

Adds the language setting to the command buffer.
Encodes character string specified with addText (p.78) according to the specified language setting in this API.

Syntax

```
public void addTextLang(int lang) throws EposException
```

Parameter

- lang : Specifies the target language.

Setting	Language
Printer.LANG_EN(default)	English (ANK)
Printer.LANG_JA	Japanese (International character set changes to Japan.)
Printer.LANG_KO	Korean (International character set changes to Korean.)
Printer.LANG_ZH_CN	Simplified Chinese (International character set changes to China.)
Printer.LANG_ZH_TW	Traditional Chinese
Printer.LANG_TH	Thai (Only for South Asia model.)

For printable character code, refer to the Technical Reference Guide of your printer.

Depending on language specification, a part of characters is printed as follows.

Language	Characters $(U+0024)	Characters \(U+005C)
Japanese	$	¥
Korean	$	₩
Simplified Chinese	¥	\
Traditional Chinese	$	\

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addTextFont

Adds the text font setting to the command buffer.

Syntax

```
public void addTextFont(int font) throws EposException
```

Parameter

- font : Specifies the font.

Setting	Language
Printer.FONT_A (default)	Font A
Printer.FONT_B	Font B
Printer.FONT_C	Font C
Printer.FONT_D	Font D
Printer.FONT_E	Font E

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addTextSmooth

Adds the smoothing setting to the command buffer.

Syntax

```
public void addTextSmooth(int smooth)
                          throws EposException
```

Parameter

- smooth : Specifies whether to enable smoothing.

Setting	Description
Printer.TRUE	Specifies smoothing.
Printer.FALSE (default)	Cancels smoothing

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addTextDouble

Adds the double-sized text setting to the command buffer.

Syntax

```
public void addTextDouble(int dw, int dh)
                          throws EposException
```

Parameter

- dw : Specifies the double-sized width.

Setting	Description
Printer.TRUE	Specifies the double-sized width.
Printer.FALSE (default)	Cancels the double-sized width
Printer.PARAM_UNSPECIFIED	Retains the current setting for double-sized width.

- dh : Specifies the double-sized height.

Setting	Description
Printer.TRUE/1	Specifies the double-sized height
Printer.FALSE/0 (default)	Cancels the double-sized height
Printer.PARAM_UNSPECIFIED	Retains the current setting for double-sized height

- When Printer.TRUE is set for both the dw and dh parameters, double width and height characters are printed.
- If Printer.PARAM_UNSPECIFIED is specified for all parameters, this returns EposException.ERR_PARAM as the return value.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addTextSize

Adds the text scale setting to the command buffer.

Syntax

```
public void addTextSize(int width, int height)
                        throws EposException
```

Parameter

- width : Specifies the horizontal scale of text.

Setting	Description
Integer from 1 to 8	Horizontal scale (default : 1)
Printer.PARAM_UNSPECIFIED	Retains the current setting for the horizontal scale.

- height : Specifies the vertical scale of text.

Setting	Description
Integer from 1 to 8	Vertical scale (default : 1)
Printer.PARAM_UNSPECIFIED	Retains the current setting for the vertical scale.

If Printer.PARAM_UNSPECIFIED is specified for all parameters, this returns EposException.ERR_PARAM as the return value.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addTextStyle

Adds the text style setting to the command buffer.

Syntax

```
public void addTextStyle(int reverse, int ul, int em,
                         int color) throws EposException
```

Parameter

- reverse : Specifies inversion of black and white for text.

Setting	Description
Printer.TRUE	Specifies the inversion of black and white parts of characters.
Printer.FALSE (default)	Cancels the inversion of black and white parts of characters.
Printer.PARAM_UNSPECIFIED	Retains the current setting for inversion of black and white.

- ul : Specifies the underline style.

Setting	Description
Printer.TRUE	Specifies underlining.
Printer.FALSE (default)	Cancels underlining.
Printer.PARAM_UNSPECIFIED	Retains the current underlining setting.

- em : Specifies the bold style.

Setting	Description
Printer.TRUE	Specifies emphasized printing of characters.
Printer.FALSE (default)	Cancels emphasized printing of characters.
Printer.PARAM_UNSPECIFIED	Retains the current setting for emphasized printing.

- color : Specifies the color.

Setting	Description
Printer.COLOR_NONE	Characters are not printed.
Printer.COLOR_1 (default)	First color
Printer.COLOR_2	Second color
Printer.COLOR_3	Third color
Printer.COLOR_4	Fourth color
Printer.PARAM_UNSPECIFIED	Retains the current color setting

If Printer.PARAM_UNSPECIFIED is specified for all parameters, this returns EposException.ERR_PARAM as the return value.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addTextPosition

Adds the horizontal print start position of text to the command buffer.

Syntax

```
public void addTextPosition(int x) throws EposException
```

Parameter

- x : Specifies the horizontal print start position (in dots).
  Specifies an integer from 0 to 65,535.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addTextVPosition

Adds the vertical print start position of text to the command buffer.

Use this API function by inserting it between addPageBegin (p.104) to addPageEnd (p.104).

Syntax

```
public void addTextVPosition(int y) throws EposException
```

Parameter

- y : Specifies the vertical print start position (in dots).
  Specifies an integer from 0 to 65535.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addFeedUnit

Adds paper feeding in dots to the command buffer.

Syntax

```
public void addFeedUnit(int unit) throws EposException
```

Parameter

- unit : Specifies the paper feed space (in dots). Specifies an integer from 0 to 255.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addFeedLine

Adds paper feeding in lines to the command buffer.

Syntax

```
public void addFeedLine(int line) throws EposException
```

Parameter

- line : Specifies the paper feed space (in lines). Specifies an integer from 0 to 255.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addFeedPosition

Adds label / black mark paper feeding to the command buffer.

- Control of label paper/black mark paper must be done in the standard mode.
- In the page mode, this API specification cannot be used.

Syntax

```
public void addFeedPosition(int position)
                        throws EposException
```

Parameter

- position : Specifies the feed position.

Setting	Description
Printer.FEED_PEELING	Feeds to the peeling position.
Printer.FEED_CUTTING	Feeds to the cutting position.
Printer.FEED_CURRENT_TOF	Feeds to the top of the current label.
Printer.FEED_NEXT_TOF	Feeds to the top of the next label.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addFeed

Adds a line feed to the command buffer.

Syntax

```
Public void addFeed() throws EposException
```

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addImage

Adds raster image printing to the command buffer. Prints the graphic rendered by the android.graphics.Bitmap class.
In the android.graphics.Bitmap class graphic, changes the specified range to binary value with dither processing, and converts it to raster image data. One pixel in an image equals to one printer dot. When an image contains any transparent color, the background color of the image is assumed to be white.

- To print a raster image at high speed, specify Printer.ALIGN_LEFT for the addTextAlign (p.76), and specify a multiple of 8 not exceeding the printer's paper width for the width parameter of this API.
- In page mode, a raster image is laid out in the current print position with the reference point being its bottom left dot. The print position will not move.

Page mode does not support printing in multiple tones. Multiple tone graphics can only be printed in standard mode.

Syntax

```
public void addImage(Bitmap data, int x, int y,
                     int width, int height, int color,
                     int mode) throws EposException
```

Parameter

- data : Specifies an instance of the android.graphics.Bitmap class.
- x : Specifies the horizontal start position in the print area.
  Specifies an integer from 0 to 65535.
- y : Specifies the vertical start position in the print area.
  Specifies an integer from 0 to 65,535.
- width : Specifies the width of the print area. Specifies an integer from 1 to 65,535.
- height : Specifies the height of the print area. Specifies an integer from 1 to 65,535.
- color : Specifies the color.

Setting	Description
Printer.COLOR_NONE	Characters are not printed.
Printer.COLOR_1 (default)	First color
Printer.COLOR_2	Second color
Printer.COLOR_3	Third color
Printer.COLOR_4	Fourth color
Printer.PARAM_DEFAULT	First color

- mode : Specifies the color mode.

Setting	Description
Printer.MODE_MONO	Monochrome (two-tone)
Printer.MODE_GRAY16	Gray scale (16-tone)
Printer.PARAM_DEFAULT	Monochrome (two-tone)

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

When the range specified with x/y and width/height does not fit the image size specified with data, EposException.ERR_PARAM is returned as a return value.

addLogo

Adds NV logo printing to the command buffer.
Prints a logo registered in the NV memory of the printer.

- A logo is registered in the printer beforehand using the following utilities.
  * Model-dedicated Utility
  * logo registration utility (TMFLogo)
- In page mode, a logo is laid out in the current print position with the reference point being its bottom left dot.

Page mode does not support printing in multiple tones. Multiple tone graphics can only be printed in standard mode.

Syntax

```
public void addLogo(int key1, int key2)
                                throws EposException
```

Parameter

- key1 : Specifies the key code 1 of an NV logo. Specifies an integer from 0 to 255.
- key2 : Specifies the key code 2 of an NV logo. Specifies an integer from 0 to 255.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addBarcode

Adds barcode printing to the command buffer.

In page mode, a barcode is laid out in the current print position with the reference point being its bottom left dot (except for HRI).

Syntax

```
public void addBarcode(String data, int type, int hri,
                       int font, int width, int height)
                       throws EposException
```

Parameter

- data : Specifies the barcode data as a string.

Specify a string that follows the barcode standard specified by the type parameter. If the specified string does not conform to the standard, a barcode will not be printed.

Barcode type	Description
UPC-A	When an 11-digit number is specified, a check digit is automatically added. When a 12-digit number is specified, the 12th digit is processed as a check digit but the check digit is not validated.
UPC-E	Specify 0 as the first digit. Specify the manufacturer code in the digits 2 to 6. Specify (right-align) the item code in the digits 7 to 11. The number of item code digits varies depending on the manufacturer code. Specify 0s in empty digits. When an 11-digit number is specified, a check digit is automatically added. When a 12-digit number is specified, the 12th digit is processed as a check digit but the check digit is not validated.
EAN13 JAN13	When an 12-digit number is specified, a check digit is automatically added. When a 13-digit number is specified, the 12th digit is processed as a check digit but the check digit is not validated.
EAN8 JAN8	When a 7-digit number is specified, a check digit is automatically added. When an 8-digit number is specified, the 8th digit is processed as a check digit but the check digit is not validated.

Barcode type	Description
CODE39	When the first character is *, the character is processed as the start character. In other cases, a start character is automatically added.
ITF	Start and stop codes are automatically added. Check digits are not added or validated.
CODABAR	Specify a start character (A to D, a to d). Specify a stop character (A to D, a to d). Check digits are not added or validated.
CODE93	Start and stop characters are automatically added. A check digit is automatically calculated and added.
CODE128	Specify a start character (CODE A, CODE B, CODE C). A stop character is automatically added. A check digit is automatically calculated and added. To encode each of the following characters, specify two characters starting with the character "{": FNC1: {1 FNC2: {2 FNC3: {3 FNC4: {4 CODE A: {A CODE B: {B CODE C: {C SHIFT: {S {: {{
GS1-128	A start character, FNC1, a check digit, and a stop character are automatically added. To automatically calculate and add a check digit for an application identifier (AI) and the subsequent data, specify the character "*" in the position of the check digit. You can enclose an application identifier (AI) in parentheses. The parentheses are used as HRI print characters and are not encoded as data. You can insert spaces between an application identifier (AI) and data. The spaces are used as HRI print characters and are not encoded as data. To encode each of the following characters, specify two characters starting with the character "{": FNC1: {1 FNC3: {3 (: {( ): {) *: {* {: {{

Barcode type	Description
GS1 DataBar Omnidirectional	Specify a 13-digit global trade item number (GTIN) not including an application identifier (AI) or a check digit.
GS1 DataBar Truncated	
GS1 DataBar Limited	
GS1 Databar Expanded	You can enclose an application identifier (AI) in parentheses. The parentheses are used as HRI print characters and are not encoded as data. To encode each of the following characters, specify two characters starting with the character "{": FNC1: {1 (: {( ): {)

To specify binary data that cannot be represented by character strings, use the following escape sequences.

String	Description
\xnn	Control code
\\	Back slash

- type : Specifies the barcode type.

Setting	Barcode type
Printer.BARCODE_UPC_A	UPC-A
Printer.BARCODE_UPC_E	UPC-E
Printer.BARCODE_EAN13	EAN13
Printer.BARCODE_JAN13	JAN13
Printer.BARCODE_EAN8	EAN8
Printer.BARCODE_JAN8	JAN8
Printer.BARCODE_CODE39	CODE39
Printer.BARCODE_ITF	ITF
Printer.BARCODE_CODABAR	CODABAR
Printer.BARCODE_CODE93	CODE93
Printer.BARCODE_CODE128	CODE128
Printer.BARCODE_GS1_128	GS1-128
Printer.BARCODE_GS1_DATABAR_OMNIDIRECTIONAL	GS1 DataBar Omnidirectional
Printer.BARCODE_GS1_DATABAR_TRUNCATED	GS1 DataBar Truncated
Printer.BARCODE_GS1_DATABAR_LIMITED	GS1 DataBar Limited
Printer.BARCODE_GS1_DATABAR_EXPANDED	GS1 Databar Expanded

- hri : Specifies the HRI position.

Setting	Description
Printer.HRI_NONE (default)	HRI not printed
Printer.HRI_ABOVE	Above the barcode
Printer.HRI_BELOW	Below the barcode
Printer.HRI_BOTH	Both above and below the barcode

- font : Specifies the HRI font.

Setting	Language
Printer.FONT_A(default)	Font A
Printer.FONT_B	Font B
Printer.FONT_C	Font C
Printer.FONT_D	Font D
Printer.FONT_E	Font E

- width : Specifies the width of each module in dots. Specifies an integer from 2 to 6.
- height : Specifies the barcode height in dots. Specifies an integer from 1 to 255.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addSymbol

Adds 2D-Code printing to the command buffer.

In page mode, a 2D-Code is laid out in the current print position with the reference point being its bottom left dot.

Syntax

```
public void addSymbol(String data, int type, int level,
                      int width, int height, int size)
                      throws EposException
```

Parameter

- data : Specifies 2D-Code data as a character string.

Specify a string that follows the 2D-Code standard specified by the type parameter. If the specified string does not conform to the standard, a 2D-Code will not be printed.

2D-Code type	Description
Standard PDF417	Convert the character string to the string in UTF-8, apply the escape sequence, and then encode the string. The data area can contain up to 928 code words in a maximum of 90 rows, each of which can contain up to 30 code words.
Truncated PDF417	
QR Code Model 1	Convert the character string to the string in Shift-JIS, apply the escape sequence, and then encode the string based on the data type as shown below. Number: 0 to 9 Alphanumeric character: 0 to 9, A to Z, space, $, %, *, +, -, ., /, : Kanji character: Shift-JIS value 8-bit, byte data: 0x00 to 0xff
QR Code Model 2	
Micro QR Code	

2D-Code type	Description
MaxiCode Mode 2	Convert the character string to the string in UTF-8, apply the escape sequence, and then encode the string. In Modes 2 and 3, when the first piece of data is ()>\ x1e01\x1dyy (where yy is a two-digit number), this is processed as the message header, and the subsequent data is processed as the primary message. In other cases, from the first piece of data, data is processed as the primary message. In Mode 2, specify the primary message in the following format: Postal code (1- to 9-digit number) GS:(\x1d) ISO country code (1- to 3-digit number) GS:(\x1d) Service class code (1- to 3-digit number) In Mode 3, specify the primary message in the following format: Postal code (1 to 6 pieces of data convertible by Code Set A) GS:(\x1d) ISO country code (1- to 3-digit number) GS:(\x1d) Service class code (1- to 3-digit number)
MaxiCode Mode 3	
MaxiCode Mode 4	
MaxiCode Mode 5	
MaxiCode Mode 6	
GS1 DataBar Stacked	Convert the character string to the string in UTF-8, apply the escape sequence, and then encode the string. Specify a 13-digit global trade item number (GTIN) not including an application identifier (AI) or a check digit.
GS1 DataBar Stacked Omnidirectional	
GS1 DataBar Expanded Stacked	Convert the character string to the string in UTF-8, apply the escape sequence, and then encode the string. You can enclose an application identifier (AI) in parentheses. The parentheses are used as HRI print characters and are not encoded as data. To encode each of the following characters, specify two characters starting with the character "{": FNC1: {1 (: {( ): {)
Aztec Code	After converting the character string to UTF-8, conduct the escape sequence and encode.
DataMatrix	After converting the character string to UTF-8, conduct the escape sequence and encode.

To specify binary data that cannot be represented by character strings, use the following escape sequences.

String	Description
\xnn	Control code
\\	Back slash

- type : Specifies the 2D-Code type.

Setting	2D-Code type
Printer.SYMBOL_PDF417_STANDARD	Standard PDF417
Printer.SYMBOL_PDF417_TRUNCATED	Truncated PDF417
Printer.SYMBOL_QRCODE_MODEL_1	QR Code Model 1
Printer.SYMBOL_QRCODE_MODEL_2	QR Code Model 2
Printer.SYMBOL_QRCODE_MICRO *	Micro QR Code
Printer.SYMBOL_MAXICODE_MODE_2	MaxiCode Mode 2
Printer.SYMBOL_MAXICODE_MODE_3	MaxiCode Mode 3
Printer.SYMBOL_MAXICODE_MODE_4	MaxiCode Mode 4
Printer.SYMBOL_MAXICODE_MODE_5	MaxiCode Mode 5
Printer.SYMBOL_MAXICODE_MODE_6	MaxiCode Mode 6
Printer.SYMBOL_GS1_DATABAR_STACKED	GS1 DataBar Stacked
Printer.SYMBOL_GS1_DATABAR_STACKED _OMNIDIRECTIONAL	GS1 DataBar Stacked Omnidirectional
Printer.SYMBOL_GS1_DATABAR_EXPANDED _STACKED	GS1 DataBar Expanded Stacked
Printer.SYMBOL_AZTECCODE_FULLRANGE	Aztec Code Full-Range mode
Printer.SYMBOL_AZTECCODE_COMPACT	Aztec Code Compact mode
Printer.SYMBOL_DATAMATRIX_SQUARE	DataMatrix ECC200 square
Printer.SYMBOL_DATAMATRIX_RECTANGLE_8	DataMatrix ECC200 rectangle, 8 lines
Printer.SYMBOL_DATAMATRIX_RECTANGLE_12	DataMatrix ECC200 rectangle, 12 lines
Printer.SYMBOL_DATAMATRIX_RECTANGLE_16	DataMatrix ECC200 rectangle, 16 lines

* TM-i firmware Ver.4.1 or later versions supported

- level : Specifies the error correction level.

Setting	Description
Printer.LEVEL_0	PDF417 error correction level 0
Printer.LEVEL_1	PDF417 error correction level 1
Printer.LEVEL_2	PDF417 error correction level 2
Printer.LEVEL_3	PDF417 error correction level 3
Printer.LEVEL_4	PDF417 error correction level 4
Printer.LEVEL_5	PDF417 error correction level 5
Printer.LEVEL_6	PDF417 error correction level 6
Printer.LEVEL_7	PDF417 error correction level 7
Printer.LEVEL_8	PDF417 error correction level 8
Printer.LEVEL_L	QR Code error correction level L
Printer.LEVEL_M	QR Code error correction level M
Printer.LEVEL_Q	QR Code error correction level Q
Printer.LEVEL_H	QR Code error correction level H
Integer from 5 to 95	Aztec Code error correction level (Default: 23)
Printer.LEVEL_DEFAULT	Default level

- Select the level according to the 2D-Code type.
- For MaxiCode and two-dimensional GS1 DataBar, select Printer.LEVEL_DEFAULT.
- Micro QR Code does not support Printer.LEVEL_H.

- width : Specifies the module width. Specifies an integer from 0 to 255.

2D-Code type	Valid value range	Default value
PDF417	2 to 8	3
QR Code	3 to 16	3
MaxiCode	Ignored	
2D GS1 Databar	2 to 8	2
Aztec Code	2 to 16	3
DataMatrix	2 to 16	3

- height : Specifies the module height. Specifies an integer from 0 to 255.

2D-Code type	Valid value range	Default value
PDF417	2 to 8 (Magnification for width)	3
QR Code	Ignored	
MaxiCode		
2D GS1 Databar		
Aztec Code		
DataMatrix		

- size : Specifies the 2D-Code maximum size. Specifies an integer from 0 to 65,535.

2D-Code type	Default value	Description
PDF417	0 (Auto)	Specifies the number of code words for each row
QR Code	Ignored	
MaxiCode		
2D GS1 Databar	0 (Auto)	Specifies the maximum width for the barcode (106 or above)
Aztec Code	Ignored	
DataMatrix		
(Others)	Ignored	

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addHLine

Adds horizontal line printing to the command buffer.
Draws horizontal lines.

> Not available in page mode.

Syntax

```
public void addHLine(int x1, int x2, int style)
                                throws EposException
```

Parameter

- x1: Specifies the start position of the horizontal line (in dots).
  Specifies an integer from 0 to 65,535.
- x2 : Specifies the end position of the horizontal line (in dots).
  Specifies an integer from 0 to 65,535.
- style : Specifies the line type.

Setting	Description
Printer.LINE_THIN	Solid line: Thin
Printer.LINE_MEDIUM	Solid line: Medium
Printer.LINE_THICK	Solid line: Thick
Printer.LINE_THIN_DOUBLE	Double line: Thin
Printer.LINE_MEDIUM_DOUBLE	Double line: Medium
Printer.LINE_THICK_DOUBLE	Double line: Thick
Printer.PARAM_DEFAULT	Solid line: Thin

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addVLineBegin

Adds the beginning of vertical line to the command buffer. Starts to draw vertical lines.

Not available in page mode.

Vertical lines are drawn until their end is specified by addVLineEnd (p.103). Use this API function with addVLineEnd.

Syntax

```
public void addVLineBegin(int x, int style)
                                throws EposException
```

Parameter

- x : Specifies the start position of the vertical line (in dots).
  Specifies an integer from 0 to 65535.
- style : Specifies the line type.

Setting	Description
Printer.LINE_THIN	Solid line: Thin
Printer.LINE_MEDIUM	Solid line: Medium
Printer.LINE_THICK	Solid line: Thick
Printer.LINE_THIN_DOUBLE	Double line: Thin
Printer.LINE_MEDIUM_DOUBLE	Double line: Medium
Printer.LINE_THICK_DOUBLE	Double line: Thick
Printer.PARAM_DEFAULT	Solid line: Thin

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addVLineEnd

Adds the end of vertical line to the command buffer. Finishes drawing vertical lines.

Not available in page mode.

Use this API function with addVLineBegin (p.102).

Syntax

```
public void addVLineEnd(int x, int style)
                        throws EposException
```

Parameter

- x : Specifies the end position of the vertical line (in dots). Specifies an integer from 0 to 65535.
- style : Specifies the type of the line you want to finish drawing.

Setting	Description
Printer.LINE_THIN	Solid line: Thin
Printer.LINE_MEDIUM	Solid line: Medium
Printer.LINE_THICK	Solid line: Thick
Printer.LINE_THIN_DOUBLE	Double line: Thin
Printer.LINE_MEDIUM_DOUBLE	Double line: Medium
Printer.LINE_THICK_DOUBLE	Double line: Thick
Printer.PARAM_DEFAULT	Solid line: Thin

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addPageBegin

Adds the switching to page mode to the command buffer. The page mode process starts.

	If using addImage (p.89) or addLogo (p.91) in page mode, printing in multiple tones is not supported.
	Vertical lines are processed in page mode until their end is specified by addPageEnd (p.104). Use this API function with PageEnd.

Syntax

```
public void addPageBegin() throws EposException
```

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addPageEnd

Adds the end of page mode to the command buffer. The page mode process ends.

	Use this API function with addPageBegin (p.104).

Syntax

```
public void addPageEnd() throws EposException
```

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addPageArea

Adds the print area in page mode to the command buffer.
Specifies the print area in page mode (coordinates). After this API function, specify a print data API function such as the addText (p.78).

	Specify a print area to cover the content to be printed. If the print data extends beyond the print area, the print result will be such that the print data has been printed incompletely.
	Use this API function by inserting it between addPageBegin (p.104) and addPageEnd (p.104).

Syntax

```
public void addPageArea(int x, int y, int width,
                        int height) throws EposException
```

Parameter

- x : Specifies the origin of the horizontal axis (in dots).
  Specifies an integer from 0 to 65,535. 0 is the left end of the printer's printable area.
- y : Specifies the origin of the vertical axis (in dots).
  Specifies an integer from 0 to 65,535. 0 is the position in which no paper feed has been performed.
- width : Specifies the width of the print area (in dots). Specifies an integer from 0 to 65,535.
- height : Specifies the height of the print area (in dots). Specifies an integer from 0 to 65,535.

	Determine the width and height of the print area according to the print direction setting. Otherwise, the print data might not be printed completely.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addPageDirection

Adds the page mode print direction setting to the command buffer. Specifies the print direction in page mode. This function can be omitted if rotation is not required.

Use this API function by inserting it between addPageBegin (p.104) and addPageEnd (p.104).

Syntax

```
public void addPageDirection(int dir)
                              throws EposException
```

Parameter

- dir : Specifies the print direction in page mode.

Setting	Description
Printer.DIRECTION_LEFT_TO_RIGHT (default)	Left to right (No rotation.Data is printed from the top left corner to the right.)
Printer.DIRECTION_BOTTOM_TO_TOP	Bottom to top (Counterclockwise rotation by 90 degrees. Data is printed from the bottom left corner to the top.)
Printer.DIRECTION_RIGHT_TO_LEFT	Right to left (Rotation by 180 degrees.Data is printed from the bottom right corner to the left.)
Printer.DIRECTION_TOP_TO_BOTTOM	Top to bottom (Clockwise rotation by 90 degrees. Data is printed from the top right corner to the bottom.)

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addPagePosition

Adds the page mode print-position-set area to the command buffer.
Specifies the print start position (coordinates) in the area specified by the addPageArea (p.105).

Use this API function by inserting it between addPageBegin (p.104) and addPageEnd (p.104).

Syntax

```
public void addPagePosition(int x, int y)
                            throws EposException
```

Parameter

- x : Specifies the horizontal print position (in dots). Specifies an integer from 0 to 65,535.
- y : Specifies the vertical print position (in dots). Specifies an integer from 0 to 65,535.

Specify the print start position (coordinates) according to the content to be printed. Refer to the following.

* To print a character string:
  Specify the left end of the baseline for the first character. This can be omitted for left-aligned printing of standard-sized characters. To print double-sized height characters, specify a value equal to or greater than 42 for y.
* To print a barcode:
  Specify the bottom left of the symbol. And specify the barcode height for y.
* To print a graphic/logo:
  Specify the bottom left of the graphic data. And specify the graphic data height for y.
* To print a 2D-Code:
  Specify the top left of the symbol. This can be omitted when printing from the top left.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addPageLine

Adds line drawing in page mode to the command buffer. Draws a line in page mode.

Diagonal lines cannot be drawn.

Use this API function by inserting it between addPageBegin (p.104) and addPageEnd (p.104).

Syntax

```
public void addPageLine(int x1, int y1, int x2, int y2,
                        int style) throws EposException
```

Parameter

- x1: Specifies the horizontal start position of the line (in dots).
  Specifies an integer from 0 to 65,535.
- y1: Specifies the vertical start position of the line (in dots).
  Specifies an integer from 0 to 65,535.
- x2 : Specifies the horizontal end position of the line (in dots).
  Specifies an integer from 0 to 65,535.
- y2 : Specifies the vertical end position of the line (in dots).
  Specifies an integer from 0 to 65,535.
- style : Specifies the line type.

Setting	Description
Printer.LINE_THIN	Solid line: Thin
Printer.LINE_MEDIUM	Solid line: Medium
Printer.LINE_THICK	Solid line: Thick
Printer.LINE_THIN_DOUBLE	Double line: Thin
Printer.LINE_MEDIUM_DOUBLE	Double line: Medium
Printer.LINE_THICK_DOUBLE	Double line: Thick
Printer.PARAM_DEFAULT	Solid line: Thin

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addPageRectangle

Adds rectangle drawing in page mode to the command buffer. Draws a rectangle in page mode.

Use this API function by inserting it between addPageBegin (p.104) and addPageEnd (p.104).

Syntax

```
public void addPageRectangle
              (int x1, int y1, int x2, int y2, int style)
               throws EposException
```

Parameter

- x1: Specifies the horizontal start position of the line (in dots).
  Specifies an integer from 0 to 65535.
- y1: Specifies the vertical start position of the line (in dots).
  Specifies an integer from 0 to 65535.
- x2 : Specifies the horizontal end position of the line (in dots).
  Specifies an integer from 0 to 65535.
- y2 : Specifies the vertical end position of the line (in dots).
  Specifies an integer from 0 to 65535.
- style : Specifies the line type.

Setting	Description
Printer.LINE_THIN	Solid line: Thin
Printer.LINE_MEDIUM	Solid line: Medium
Printer.LINE_THICK	Solid line: Thick
Printer.LINE_THIN_DOUBLE	Double line: Thin
Printer.LINE_MEDIUM_DOUBLE	Double line: Medium
Printer.LINE_THICK_DOUBLE	Double line: Thick
Printer.PARAM_DEFAULT	Solid line: Thin

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addCut

Adds paper cut to the command buffer. Sets paper cut.

> Not available in page mode.

Syntax

```
public void addCut(int type) throws EposException
```

Parameter

- type : Specifies the paper cut type.

Setting	Description
Printer.CUT_NO_FEED	Cut without feeding (The paper is cut without being fed.)
Printer.CUT_FEED	Feed cut (The paper is fed to the cut position and then is cut.)
Printer.CUT_RESERVE	Cut reservation (Printing continues until the cut position is reached, at which the paper is cut.)
Printer.PARAM_DEFAULT	Feed cut (The paper is fed to the cut position and then is cut.)

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addPulse

Adds the drawer kick to the command buffer. Sets the drawer kick.

- Not available in page mode.
- The drawer and the buzzer cannot be used together.

Syntax

```
public void addPulse(int drawer, int time)
                        throws EposException
```

Parameter

- drawer : Specifies the drawer kick connector.

Setting	Description
Printer.DRAWER_1	Pin 2 of the drawer kick-out connector
Printer.DRAWER_2	Pin 5 of the drawer kick-out connector
Printer.PARAM_DEFAULT	Pin 2 of the drawer kick-out connector

- time : Specifies the ON time of the drawer kick signal.

Setting	Description
Printer.PULSE_100	100 ms
Printer.PULSE_200	200 ms
Printer.PULSE_300	300 ms
Printer.PULSE_400	400 ms
Printer.PULSE_500	500 ms
Printer.PARAM_DEFAULT	100 ms

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addSound

Adds the turning on of the buzzer to the command buffer. Sets the buzzer.

- Not available in page mode.
- The buzzer function and the drawer cannot be used together.
- This API function cannot be used if the printer is not provided with the buzzer.

Syntax

```
public void addSound(int pattern, int repeat, int cycle)
                                    throws EposException
```

Parameter

- pattern : Specifies the buzzer pattern.

Setting	Description
Printer.PATTERN_NONE	Stop
Printer.PATTERN_A	Pattern A (Optional Buzzer)
Printer.PATTERN_B	Pattern B (Optional Buzzer)
Printer.PATTERN_C	Pattern C (Optional Buzzer)
Printer.PATTERN_D	Pattern D (Optional Buzzer)
Printer.PATTERN_E	Pattern E (Optional Buzzer)
Printer.PATTERN_ERROR	Error sound pattern (Optional Buzzer)
Printer.PATTERN_PAPER_END	Pattern when there is no paper (Optional Buzzer)
Printer.PATERN_1	Pattern 1 (Internal Buzzer)
Printer.PATERN_2	Pattern 2 (Internal Buzzer)
Printer.PATERN_3	Pattern 3 (Internal Buzzer)
Printer.PATERN_4	Pattern 4 (Internal Buzzer)
Printer.PATERN_5	Pattern 5 (Internal Buzzer)
Printer.PATERN_6	Pattern 6 (Internal Buzzer)
Printer.PATERN_7	Pattern 7 (Internal Buzzer)
Printer.PATERN_8	Pattern 8 (Internal Buzzer)
Printer.PATERN_9	Pattern 9 (Internal Buzzer)
Printer.PATERN_10	Pattern 10 (Internal Buzzer)
Printer.PARAM_DEFAULT	Pattern A

- repeat : Specifies the number of repeats.

Setting	Description
0	The buzzer does not stop.
1 to 255	Number of repeats
Printer.PARAM_DEFAULT	One time

After “0” is specified for repeat, if you want to stop the buzzer, execute this API function and specify Printer.PATTERN_NONE for pattern.

- cycle: Specifies the buzzer sounding cycle (in units of milliseconds)
  Specifies an integer from 1,000 to 25,500.

A sound of the buzzer is valid in Printer.PATTERN_1 to Printer.PATTERN_10.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addLayout

Adds paper layout information to the command buffer.

> Setting of page layout must be done in the standard mode. In the page mode, addLayout cannot be specified.

Syntax

```
public void addLayout(int type, int width, int height,
                      int marginTop, int marginBottom,
                      int offsetCut, int offsetLabel)
                      throws EposException
```

Parameter

- type : Specifies the paper type.

Setting	Description
Printer.LAYOUT_RECEIPT	Receipt (without black mark)
Printer.LAYOUT_RECEIPT_BM	Receipt (with black mark)
Printer.LAYOUT_LABEL	Die-cut label (without black mark)
Printer.LAYOUT_LABEL_BM	Die-cut label (with black mark)

- width : Specifies paper width (in units of 0.1 mm). Specifies an integer from 290 to 600.
- height : Specifies paper height (in units of 0.1 mm).

Paper Type	Valid value range	Description
Receipt (without black mark)	0	Setup not necessary
Receipt (with black mark)	0 (Auto) 284 to 1550 (Manual)	Distance from the top of black mark to the top of next black mark
Die-cut label (without black mark)		Distance from the top of label to the top of next label
Die-cut label (with black mark)		Distance from the bottom of black mark to the bottom of next black mark.

- margin_top : Specifies top margin (in units of 0.1 mm).

Paper Type	Valid value range	Description
Receipt (without black mark)	0	Setup not necessary
Receipt (with black mark)	-150 to 1500	Distance from the top of black mark
Die-cut label (without black mark)	0 to 1500	Distance from the top of label
Die-cut label (with black mark)	-15 to 1500	Distance from the bottom of black mark

- margin_bottom :

  Specifies bottom margin (in units of 0.1 mm).

Paper Type	Valid value range	Description
Receipt (without black mark)	0	Setup not necessary
Receipt (with black mark)	0	
Die-cut label (without black mark)	-15 to 0	Distance from the bottom of label (paper feed direction is a positive number)
Die-cut label (with black mark)	-15 to 15	Distance from the top of black mark (paper feed direction is a positive number)

- offset_cut : Specifies cut position (in units of 0.1 mm).
  In case of die cut label paper, it is a distance from the bottom of label.
  When a paper has black mark, it is a distance from the beginning of black mark.

Paper Type	Valid value range	Description
Receipt (without black mark)	0	Setup not necessary
Receipt (with black mark)	-290 to 50	Distance from the top of black mark to the cutting position
Die-cut label (without black mark)	0 to 50	Distance from the bottom of label to the cutting position
Die-cut label (with black mark)	0 to 50	Distance from the top of black mark to the cutting position

- offset_label : Specifies label bottom position (sd) per 0.1 mm unit.

Paper Type	Valid value range	Description
Receipt (without black mark)	0	Setup not necessary
Receipt (with black mark)	0	
Die-cut label (without black mark)	0	
Die-cut label (with black mark)	0 to 15	Distance from the top of black mark to the bottom of label

Exceptions

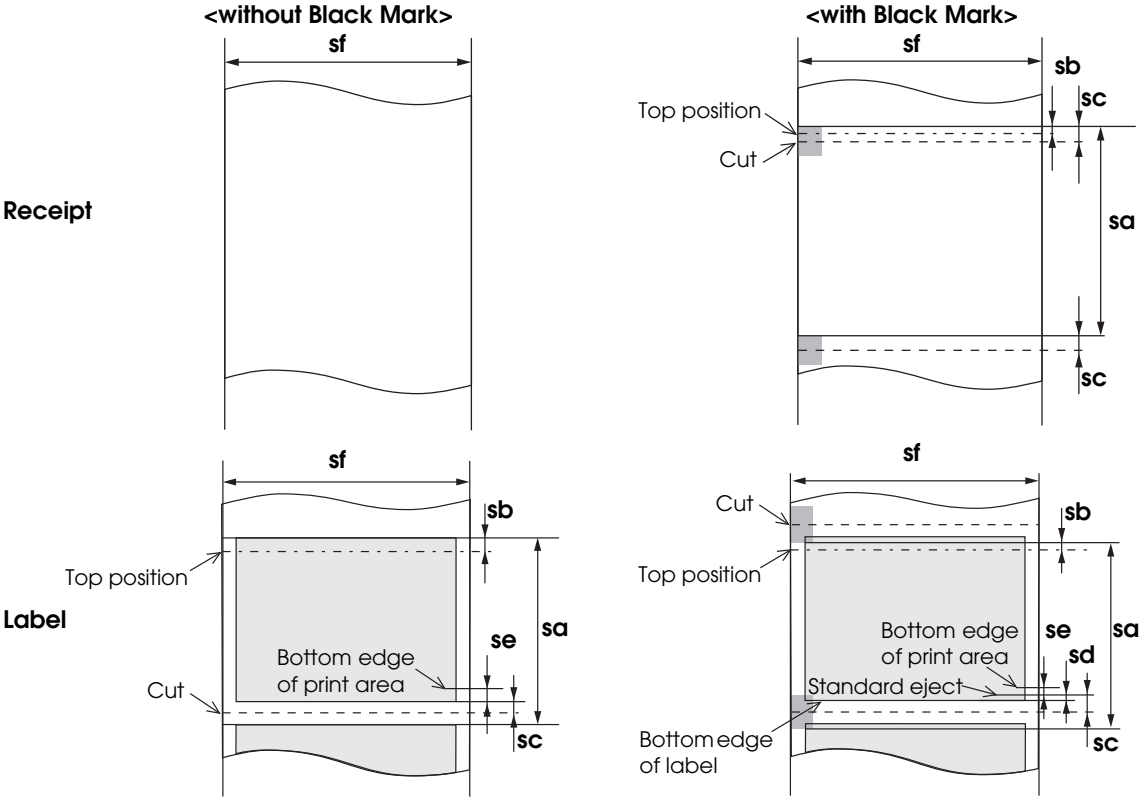
When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

Detailed description

See below for the parameters that can be specified for each type of paper, and the positions for those parameters.

Mark	Parameter
sf	width
sa	height
sb	margin_top
se	margin_bottom
sc	offset_cut
sd	offset_label



addCommand

Adds commands to the command buffer. Sends ESC/POS commands.

For information about ESC/POS commands, see the ESC/POS Command Reference.
You can view the ESC/POS Command Reference from https://reference.epson-biz.com/.

Syntax

```
public void addCommand(byte[] data) throws EposException
```

Parameter

- data : Specifies ESC/POS command as a binary data.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addRecovery

Adds the recovery from errors to the command buffer.
This API is executed when a error that can be automatically recovered from occurs.

- If this API is included in a print command, other print commands included in the print document are not executed.
- We recommend using the recover (p.119) for this function.

Syntax

```
public void addRecovery() throws EposException
```

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

addReset

Adds the printer reset to the command buffer.

- Other printing commands in the print document are ignored.
- We recommend using the reset (p.119) for this function.

Syntax

```
public void addReset() throws EposException
```

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

startMonitor

Starts a status event notification of the printer.

Syntax

```
public void startMonitor()throws EposException
```

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

stopMonitor

Ends a status event notification of the printer.

Syntax

```
public void stopMonitor()throws EposException
```

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_FAILURE	An unspecified error occurred.

recover

The printer recovers from the error status.

This API is executed when a error that can be automatically recovered from occurs.

This API restores from error condition regardless of the setting of force property (p.122).

Syntax

```
Public void recover() throws EposException
```

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_CONNECT	Failed to communicate with a device.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.
EposException.ERR_PROCESSING	Could not execute a process.

reset

Resets the printer.

Execution of this API results in the value of the force property (p.122) as Printer.FALSE.

Syntax

```
Public void reset() throws EposException
```

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_CONNECT	Failed to communicate with a device.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.
EposException.ERR_PROCESSING	Could not execute a process.

halftone property

Halftone processing method.
Applied to monochrome (2 tones) printing of addImage (p.89).

Syntax

Setter

```
public void setHalftone(int halftone)
                                throws EposException
```

Getter

```
public int getHalftone()
```

Set Value

Setting	Description
Printer.HALFTONE_DITHER (default)	Dithering, suitable for printing graphics only.
Printer.HALFTONE_ERROR_DIFFUSION	Error diffusion, suitable for printing text and graphics together.
Printer.HALFTONE_THRESHOLD	Threshold, suitable for printing text only.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.

brightness property

Brightness correction value.
Applied when executing addImage (p.89).

Syntax

Setter

```
public void setBrightness(double brightness)
                                throws EposException
```

Getter

```
public double getBrightness()
```

Set Value

A gamma value in the range 0.1-10.0 is specified for the brightness correction value.
The default value is 1.0.

When any value except for 1.0 is specified for the brightness correction value, the printing speed gets slower.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.

force property

This is the forced transmission mode.
If you enable forced transmission mode, print commands are forcibly sent to the printer.

- Use forced transmission mode when the printer is offline. It will result in an error if the printer is online. When the printer is used online, "EposCallbackCode.ERR_PORT" is returned as the code for the response document received event (p.124).
- The following functions are enabled in forced transmission mode.
  - Drawer kick-out (addPulse (p.111))
  - Stopping the buzzer (addSound (p.112))
  - Recovery from errors that can be recovered from (addRecovery (p.117))
  - Reset (addReset (p.118))
  - Sending commands in real time (addCommand (p.117))

Syntax

Setter

```
public void setForce(boolean force) throws EposException
```

Getter

```
public boolean getForce()
```

Set Value

Setting	Description
true	Sets the forced transmission mode to valid.
false	Sets the forced transmission mode to invalid.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.

timeout property

Specifies the timeout time (ms) of the message sent with the sendData (p.74).

Syntax

Setter

```
public void setTimeout(int timeout) throws EposException
```

Getter

```
public int getTimeout()
```

Set Value

Specifies the timeout within the range of 0 to 600,000.

The default value is 10,000.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.

interval property

Specifies the interval of upgrading the status.

Syntax

Setter

```
public void setInterval(int interval)
                                throws EposException
```

Getter

```
public int getInterval()
```

Set Value

Specifies the interval to update the status within the range of 1,000 to 6,000.

The default value is 3,000.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.

setReceiveEventCallback

Registers a callback method that receives the response document received event.

- This API can be executed following execution of createDevice (p.51).
- When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setReceiveEventCallback
                        (ReceiveListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

❏ `public interface ReceiveListener extends EventListener`

❏ For acquiring the job ID

```
public interface JobReceiveListener
                        extends ReceiveListener
```

Supported in TM-i firmware Ver.4.1 or later when use the JobReceiveListener.

Listener Registration Method

❑ `void` ***`onPtrReceive`*** `(String ipAddress, String deviceId, int success, int code,int status, int battery)`

❑ For acquiring the job ID

`void` ***`onPtrReceive`*** `(String ipAddress, String deviceId, int success, int code,int status, int battery, String printJobId)`

Supported in TM-i firmware Ver.4.1 or later when use the method with a printJobId parameter.

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.
- success: Sets printing execution results.

Error status	Description
Printer.TRUE	• Printing succeeded • Processing succeeded (When the spooler function is enabled)
Printer.FALSE	• Printing Failed • Processing Failed (When the spooler function is enabled)

- code: Sets execution results of process.

Error status	Description
EposCallbackCode.SUCCESS	Processing was successful.
EposCallbackCode.PRINTING *	Print job now printing.
EposCallbackCode.ERR_AUTOMATICAL	An automatically recoverable error occurred
EposCallbackCode.ERR_BATTERY_LOW	No remaining battery
EposCallbackCode.ERR_COVER_OPEN	A cover open error occurred
EposCallbackCode.ERR_CUTTER	An autocutter error occurred
EposCallbackCode.ERR_MECHANICAL	A mechanical error occurred
EposCallbackCode.ERR_EMPTY	No paper in roll paper end sensor
EposCallbackCode.ERR_UNRECOVERABLE	An unrecoverable error occurred
EposCallbackCode.ERR_FAILURE	The request document contains a syntax error
EposCallbackCode.ERR_NOT_FOUND	The printer with the specified device ID does not exist
EposCallbackCode.ERR_SYSTEM	An error occurred on the printing system
EposCallbackCode.ERR_PORT	An error was detected on the communication port
EposCallbackCode.ERR_TIMEOUT	A print timeout occurred
EposCallbackCode.ERR_JOB_NOT_FOUND *	The specified job ID does not exist.
EposCallbackCode.ERR_SPOOLER *	There is not enough space available in the printing queue.

* TM-i firmware Ver.4.1 or later versions supported

- status: Sets printer status.

Printer status	Description
Printer.ST_NO_RESPONSE	No response from the TM printer
Printer.ST_PRINT_SUCCESS	Printing complete/Spooling complete
Printer.ST_DRAWER_KICK	Status of the 3rd pin of the drawer kick-out connector = "H"
Printer.ST_BATTERY_OFFLINE	Battery offline status
Printer.ST_OFF_LINE	Offline
Printer.ST_COVER_OPEN	The cover is open
Printer.ST_PAPER_FEED	Paper is being fed by a paper feed switch operation
Printer.ST_WAIT_ON_LINE	Waiting to be brought back online
Printer.ST_PANEL_SWITCH	The paper feed switch is being pressed (ON)
Printer.ST_MECHANICAL_ERR	A mechanical error occurred
Printer.ST_AUTOCUTTER_ERR	An autocutter error occurred
Printer.ST_UNRECOVER_ERR	An unrecoverable error occurred
Printer.ST_AUTORECOVER_ERR	An automatically recoverable error occurred
Printer.ST_RECEIPT_NEAR_END	No paper in roll paper near end sensor
Printer.ST_RECEIPT_END	No paper in roll paper end sensor
Printer.ST_BUZZER	A buzzer is on (only for applicable devices)
Printer.ST_SPOOLER_IS_STOPPED *	The spooler has stopped

* TM-i firmware Ver.4.1 or later versions supported

- battery: Sets battery status.
  Status of power

Battery status	Description
0x30XX	The AC adapter is connected
0x31XX	The AC adapter is connected

Remaining battery

Battery status	Description
0xXX36	Battery amount 6
0xXX35	Battery amount 5
0xXX34	Battery amount 4
0xXX33	Battery amount 3
0xXX32	Battery amount 2
0xXX31	Battery amount 1 (Near end)
0xXX30	Battery amount 0 (Real end)

0 is shown when the model doesn't have a battery installed.

- printJobId: Sets the print job ID.

setStatusChangeEventCallback

This registers a callback method that receives StatusChangeEvent.

- This API can be executed following execution of createDevice (p.51).
- When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setStatusChangeEventCallback
                        (StatusChangeListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface StatusChangeListener
                 extends EventListener
```

Listener Registration Method

```
void onPtrStatusChange(String ipAddress,
                       String deviceId, int status)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.
- status: Sets printer status.

setBatteryStatusChangeEventCallback

This registers a callback method that receives BatteryStatusChangeEvent.

- This API can be executed following execution of createDevice (p.51).
- When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setBatteryStatusChangeEventCallback
                    (BatteryStatusChangeListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface BatteryStatusChangeListener
                 extends EventListener
```

Listener Registration Method

```
void onPtrBatteryStatusChange
                    (String ipAddress, String deviceId,
                     int battery)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.
- battery: Sets battery status.

setOnlineEventCallback

This registers a callback method that receives OnlineEvent.

- This API can be executed following execution of createDevice (p.51).
- When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setOnlineEventCallback
                          (OnlineListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface OnlineListener extends EventListener
```

Listener Registration Method

```
void onPtrOnline(String ipAddress, String deviceId)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.

setOfflineEventCallback

This registers a callback method that receives OfflineEvent.

- This API can be executed following execution of createDevice (p.51).
- When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setOfflineEventCallback
                        (OfflineListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface OfflineListener extends EventListener
```

Listener Registration Method

```
void onPtrOffline(String ipAddress, String deviceId)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.

setPowerOffEventCallback

This registers a callback method that receives PowerOffEvent.

- This API can be executed following execution of createDevice (p.51).
- When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setPowerOffEventCallback
                        (PowerOffListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface PowerOffListener extends EventListener
```

Listener Registration Method

```
void onPtrPowerOff(String ipAddress, String deviceId)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.

setCoverOkEventCallback

This registers a callback method that receives CoverOkEvent.

- This API can be executed following execution of createDevice (p.51).
- When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setCoverOkEventCallback
                        (CoverOkListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface CoverOkListener extends EventListener
```

Listener Registration Method

```
void onPtrCoverOk(String ipAddress, String deviceId)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.

setCoverOpenEventCallback

This registers a callback method that receives CoverOpenEvent.

- This API can be executed following execution of createDevice (p.51).
- When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setCoverOpenEventCallback
                        (CoverOpenListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface CoverOpenListener extends EventListener
```

Listener Registration Method

```
void onPtrCoverOpen(String ipAddress, String deviceId)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.

setPaperOkEventCallback

This registers a callback method that receives PaperOkEvent.

- This API can be executed following execution of createDevice (p.51).
- When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setPaperOkEventCallback
                        (PaperOkListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface PaperOkListener extends EventListener
```

Listener Registration Method

```
void onPtrPaperOk(String ipAddress, String deviceId)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.

setPaperNearEndEventCallback

This registers a callback method that receives PaperNearEndEvent.

- This API can be executed following execution of createDevice (p.51).
- When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setPaperNearEndEventCallback
                        (PaperNearEndListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface PaperNearEndListener
                 extends EventListener
```

Listener Registration Method

```
void onPtrPaperNearEnd(String ipAddress,
                       String deviceId)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.

setPaperEndEventCallback

This registers a callback method that receives PaperEndEvent.

- This API can be executed following execution of createDevice (p.51).
- When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setPaperEndEventCallback
                        (PaperEndListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface PaperEndListener extends EventListener
```

Listener Registration Method

```
void onPtrPaperEnd(String ipAddress, String deviceId)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.

setDrawerClosedEventCallback

This registers a callback method that receives DrawerClosedEvent.

- This API can be executed following execution of createDevice (p.51).
- When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setDrawerClosedEventCallback
                        (DrawerClosedListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface DrawerClosedListener
                 extends EventListener
```

Listener Registration Method

```
void onPtrDrawerClosed(String ipAddress,
                       String deviceId)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.

setDrawerOpenEventCallback

This registers a callback method that receives DrawerOpenEvent.

- This API can be executed following execution of createDevice (p.51).
- When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setDrawerOpenEventCallback
                        (DrawerOpenListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface DrawerOpenListener
                 extends EventListener
```

Listener Registration Method

```
void onPtrDrawerOpen(String ipAddress, String deviceId)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.

setBatteryOkEventCallback

This registers a callback method that receives BatteryOkEvent.

- This API can be executed following execution of createDevice (p.51).
- When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setBatteryOkEventCallback
                        (BatteryOkListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface BatteryOkListener extends EventListener
```

Listener Registration Method

```
void onPtrBatteryOk(String ipAddress, String deviceId)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.

setBatteryLowEventCallback

This registers a callback method that receives BatteryLowEvent.

- This API can be executed following execution of createDevice (p.51).
- When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setBatteryLowEventCallback
                        (BatteryLowListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface BatteryLowListener
                 extends EventListener
```

Listener Registration Method

```
void onPtrBatteryLow(String ipAddress, String deviceId)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.

Scanner Class

setDataEventCallback

This registers a callback method that receives DataEvent.

When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setDataEventCallback(DataListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface DataListener extends EventListener
```

Listener Registration Method

```
void onScanData(String ipAddress, String deviceId,
                String input)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.
- input: Sets detected character string.

SimpleSerial Class

sendCommand

Sends a desired command.

Syntax

```
public void sendCommand(byte[] data)
                        throws EposException
```

Parameter

- data: Specifies a command in byte array.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_CONNECT	Could not communicate.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.

setCommandReplyEventCallback

This registers a callback method that receives CommandReplyEvent.

When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setCommandReplyEventCallback
                        (CommandRerplayListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface CommandReplyListener
                  extends EventListener
```

Listener Registration Method

```
void onSimpleSerialCommandReply
                        (String ipAddress,
                         String deviceId, byte[] data)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- deviceId: Sets Device ID.
- data: Sets binary data of response data.

CommBoxManager Class

openCommBox

Opens the Communication Box.

Syntax

```
public void openCommBox(String boxId, String memberId,
                        int[] sequence,
                        OpenCommBoxListener target)
                        throws EposException
```

Parameter

- boxId: Specifies the ID of the box to open.
- memberId: Specify the member ID for identifying oneself in the Communication Box..
- sequence: Sets a process sequence number when this API is executed.
- target : This specifies the object (listener interface) that has the notification destination method.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_CONNECT	Could not communicate.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.
EposException.ERR_PROCESSING	Could not execute a process.

Listener Interface

```
public interface OpenCommBoxListener
                 extends EventListener
```

Listener Registration Method

```
void onOpenCommBox(String ipAddress,
          CommBox commBoxObject, int code, int sequence)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- commBoxObject:
  The EposCommBox object is returned for the first parameter.
  If this API fails, NULL is returned.

- code: Sets error status of processing results.

Error status	Description
EposCallbackCode.SUCCESS	Successfully opened the Communication Box.
EposException.ERR_CONNECT	Could not communicate.
EposCallbackCode. ERR_ALREADY_OPENED	The Communication Box is already opened.
EposCallbackCode. ERR_ALREADY_USED	The specified member ID is already in use.
EposCallbackCode. ERR_BOX_COUNT_OVER	The maximum number of Communication Boxes that can be made is exceeded.
EposCallbackCode. ERR_BOX_CLIENT_OVER	The maximum number that can belong to a Communication Box is exceeded.
EposCallbackCode.ERR_PARAM	Parameter error.
EposCallbackCode.ERR_FAILURE	Other error occurred.

- sequence: Sets the same sequence number with a returned value to 3rd parameter when this API is executed.

closeCommBox

Closes the Communication Box.

Syntax

```
public void closeCommBox(CommBox commBoxObject,
                         int[] sequence,
                         CloseCommBoxListener target)
                         throws EposException
```

Parameter

- commBoxObject::
  Specifies the commBox object to be closed.
- sequence: Sets a process sequence number when this API is executed.
- target : This specifies the object (listener interface) that has the notification destination method.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_CONNECT	Could not communicate.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.
EposException.ERR_PROCESSING	Could not execute a process.

Listener Interface

```
public interface CloseCommBoxListener
                 extends EventListener
```

Listener Registration Method

```
void onCloseCommBox(String ipAddress, int code,
                    int sequence)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- code: Sets error status of processing results.

Error status	Description
EposCallbackCode.SUCCESS	Successfully closed the Communication Box.
EposException. ERR_CONNECT	Could not communicate.
EposCallbackCode. ERR_NOT_OPENED	A Communication Box is not open.
EposCallbackCode. ERR_FAILURE	Other error occurred.

- sequence: Sets the same sequence number with a returned value to 2nd parameter when this API is executed.

CommBox Class

getCommHistory

Acquires the transfer history of the Communication Box.

Syntax

```
public void getCommHistory(int[] sequence,
                           GetCommHistoryListener target)
                           throws EposException
```

Parameter

- sequence: Sets a process sequence number when this API is executed.
- target : This specifies the object (listener interface) that has the notification destination method.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_CONNECT	Could not communicate.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.
EposException.ERR_PROCESSING	Could not execute a process.

Listener Interface

```
public interface GetCommHistoryListener
                 extends EventListener
```

Listener Registration Method

```
void onGetCommHistory
       (String ipAddress, int code,
        ArrayList<HashMap<String, String>> historyList,
        int sequence)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- code: Sets error status of processing results.

Error status	Description
EposCallbackCode.SUCCESS	Successfully closed the Communication Box.
EposCallbackCode. ERR_NOT_OPENED	A Communication Box is not open.
EposCallbackCode. ERR_FAILURE	Other error occurred.

- historyList: Sets a communication history of Communication Box.

historyList(index)	
Key	**Description**
senderId	Sender's memberID.
receiverId	Receiver's memberID.
message	Received message.

- sequence: Sets the same sequence number with a returned value to 1st parameter when this API is executed.

sendData

Sends a message to the Communication Box.

Syntax

```
public void sendData(String message, String memberId,
                     int[] sequence,
                     SendDataListener target)
                     throws EposException
```

Parameter

- message: Specifies the data to send to the Communication Box.
- memberId: Specifies the destination member ID.
  If NULL or " "(space) is specified, it will be sent to all members in the Communication Box except for the sender.
- sequence: Sets a process sequence number when this API is executed.
- target : This specifies the object (listener interface) that has the notification destination method.

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_CONNECT	Could not communicate.
EposException.ERR_MEMORY	Could not allocate memory.
EposException.ERR_FAILURE	An unspecified error occurred.
EposException.ERR_PROCESSING	Could not execute a process.

Listener Interface

```
public interface SendDataListener extends EventListener
```

Listener Registration Method

```
void onCommBoxSendData(String ipAddress, int code,
                       int count, int sequence)
```

Parameter

- ipAddress: Sets IP address of TM-i.

- code: Sets error status of processing results.

Error status	Description
EposCallbackCode.SUCCESS	Successfully sent the message.
EposCallbackCode. ERR_NOT_OPENED	A Communication Box is not open.
EposCallbackCode. ERR_NOT_FOUND	The specified member is not found.
EposCallbackCode. ERR_FAILURE	Other error occurred.

- count: The number of clients the message was sent to is returned as the second parameter.
- sequence: Sets the same sequence number with a returned value to 3rd parameter when this API is executed.

setReceiveEventCallback

This registers a callback method that notifies a receipt of message of Communication Box.

When this API is executed on multiple occasions, the notification destination method that is specified afterwards is overwritten.

Syntax

```
public void setReceiveEventCallback
                        (ReceiveListener target)
```

Parameter

- target : This specifies the object (listener interface) that has the notification destination method.

If NULL is specified for either the method or target, the notification destination registration is nullified.

Listener Interface

```
public interface ReceiveListener extends EventListener
```

Listener Registration Method

```
void onCommBoxReceive(String ipAddress, String senderId,
                      String receiverId, String message)
```

Parameter

- ipAddress: Sets IP address of TM-i.
- senderId: The sender's member ID is set.
  If there is no specified member ID when the openCommBox was executed, " " (space) is set.
- receiverId: The receiver's member ID is set.
  If the data was sent to all members in the Communication Box, " " (space) is set.
- message: The sent data is set.

EposDeviceLog Class

setLogSettings

Sets the log output function.

Syntax

```
public static void setLogSettings
                (Context context, int period,
                 int enabled, String ipAddress, int port,
                 int logSize, int logLevel)
                 throws EposException
```

Parameter

- period : Specifies the API of setting the log output function.

Setting	Description
Log.LOG_TEMPORARY	The settings of this API are disabled when the application is ended.
Log.LOG_PERMANENT	The settings of this API are enabled even after the application is ended.

To specify period for the Log.LOG_PERMANENT, set the permissions for the application to access the storage.

- enabled : Specifies whether to enable the log output function and the log output destination.

Setting	Description
Log.LOG_DISABLE	Disables the log output function.
Log.LOG_STORAGE	Outputs log data to the device's storage.
Log.LOG_TCP	Outputs log data over TCP.

- To specify enabled for the Log.LOG_STORAGE, set the permissions for the application to access the storage.
- To specify enabled for the Log.LOG_TCP, set the permissions for the application to access the network.

- ipAddress : Specifies the IPv4 IP address for TCP communication.

If either of the following values is specified for enabled, NULL can be specified for this parameter.

* Log.LOG_DISABLE
* Log.LOG_STORAGE

- port : Specifies the port number for TCP communication. Specifies an integer from 0 to 65,535.

Even if either of the following values is specified for enabled, specify an integer within the range.
* Log.LOG_DISABLE
* Log.LOG_STORAGE

- logSize : Specifies the maximum size of log data that is saved on the device's storage. Specifies an integer from 1 to 50 (Unit:MB).

Even if either of the following values is specified for enabled, specify an integer within the range.
* Log.LOG_DISABLE
* Log.LOG_TCP

- logLevel : Specifies the level of log data to be output.

Setting	Description
Log.LOG_LOW	Low level

Exceptions

When processing failed, EposException of the following error status is generated.
Acquire with getErrorStatus (p.156).

Error status	Description
EposException.ERR_PARAM	Invalid parameter was passed.
EposException.ERR_FAILURE	An unspecified error occurred.

EposException class

getErrorStatus

Acquires error status when an exception of API provided by the ePOS-Device SDK is thrown.

Syntax

```
public int getErrorStatus()
```

Return value

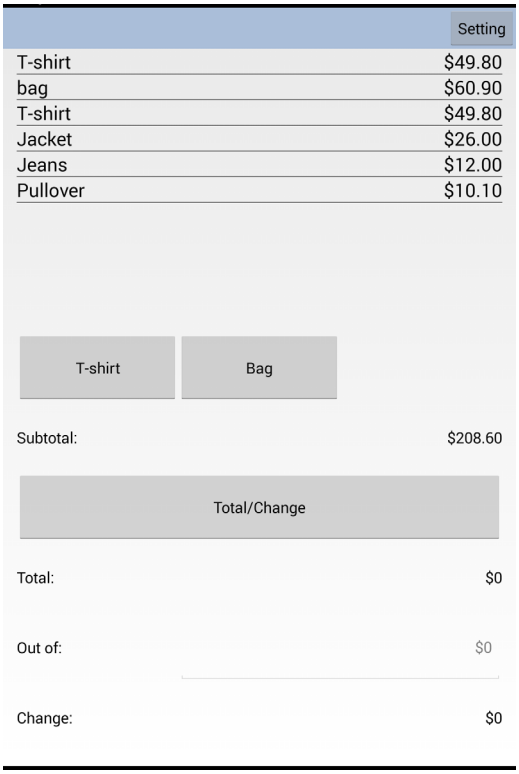
Returns error status set with API where an exception is generated.

Sample Program

This chapter describes how to use the sample program.

- The sample program is provided as an Android application project for use with Eclipse, including the Java source files.
- For an Android application for TM printers developed using ePOS-Device SDK, the following program is available.

Functionality



The Sample Program has the following functionality.

- ❏ Connecting/ Disconnecting with the TM-i
- ❏ Creation of the device object
- ❏ Reading barcode
- ❏ Displaying customer display
- ❏ Printing of receipt
- ❏ Acquiring printer status
- ❏ Error processing

Usage Environment

Development Environment

- Android SDK r22
- Java Development Kit 7
- Eclipse
- ADT Plugin for Eclipse

Required device

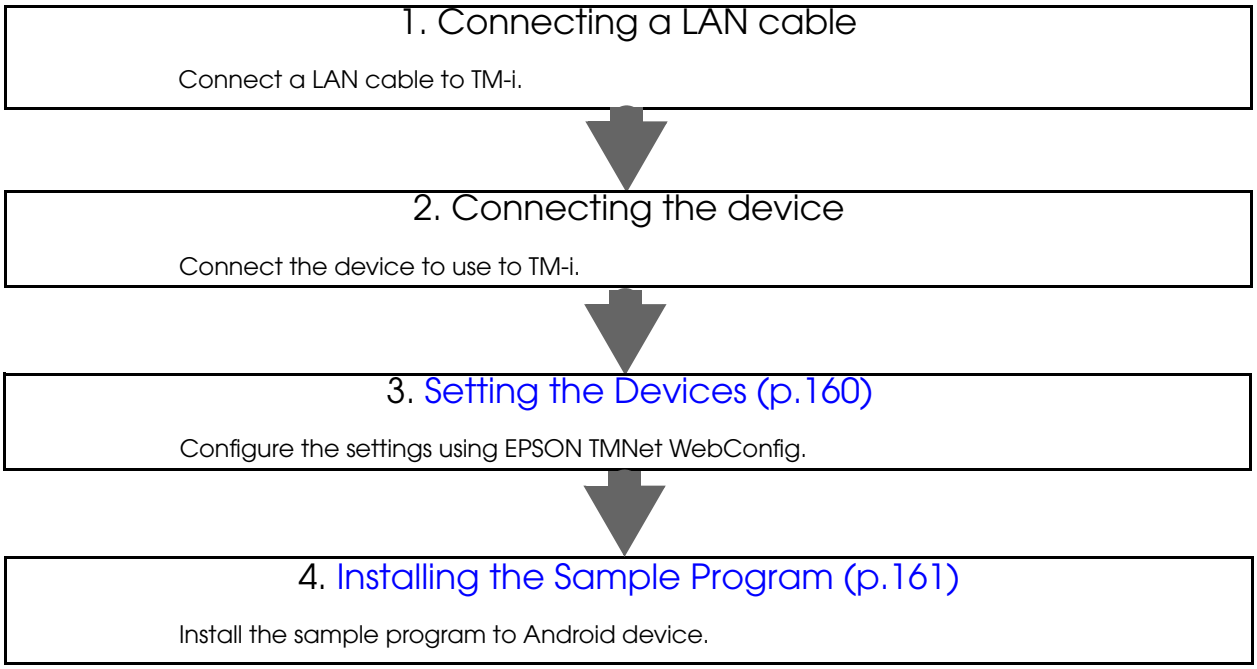
- TM-i
- Customer display
- Barcode scanner

Android Device

❑ Device that supports ARMv5TE

Environment Setting

The workflow of setting an environment for the sample program is shown as follows:


1. Connecting a LAN cable
Connect a LAN cable to TM-i.
2. Connecting the device
Connect the device to use to TM-i.
3. Setting the Devices (p.160)
Configure the settings using EPSON TMNet WebConfig.
4. Installing the Sample Program (p.161)
Install the sample program to Android device.

Setting the Devices

EPSON TMNet WebConfig can be launched from your web browser.

For details on settings used, refer to the Technical Reference Guide for each TM-i.

The following devices must be configured for the operating environment using the sample program.

❏ Customer display

Setting	Setting Value
Customer Display	Use

The device ID of the customer display is fixed to "local_display".

❏ POS keyboard

Setting	Setting Value
Device ID	local_keyboard
Device name	Select the relevant device from the list.
Control Script	Keyboard_Generic.js

❏ Barcode scanner

Setting	Setting Value
Device ID	local_scanner
Device name	Select the relevant device from the list.
Control Script	Scanner_Generic.js

Installing the Sample Program

Install the sample program to Android device following the next steps.

1 Extract the sample program zip file to a directory of your choosing.

2 In Eclipse, go to (File)-(Import), select (General)-(Existing Project into Workspace), and then click (Next).

3 The Import Projects window will be displayed. Make the settings shown below and click (Finish).

Item	Setting
Select root directory	Specify the directory where you extracted the sample program zip file.
Copy projects into workspace	Check this option.

4 In Package Explorer view, right click on the "Composite_sample" project and select (Properties).

5 The Properties for Composite_sample window will be displayed. Make the settings shown below and click (OK).

Item	Setting
Android	Select the Android OS version of the target device.
Java Build Path	In Libraries, confirm that the path to ePOS-Device.jar, which is located in the Composite_sample project that was copied into the project workspace, is set.

6 In Package Explorer view, right click on the "Composite_sample" project and select (Run As, Android Application).

7 The sample program will be installed to the target Android device, and then the program will start up.

How to Use the Sample Program

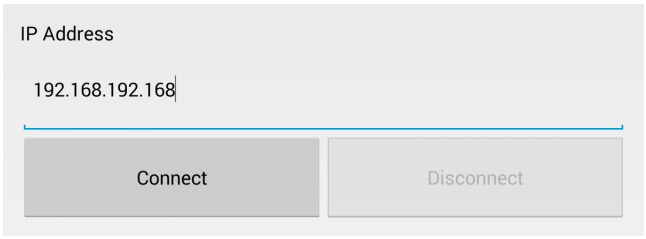
Setting

Use the sample program as follows:

1. Start the sample program. For details, refer to Environment Setting (p.159).

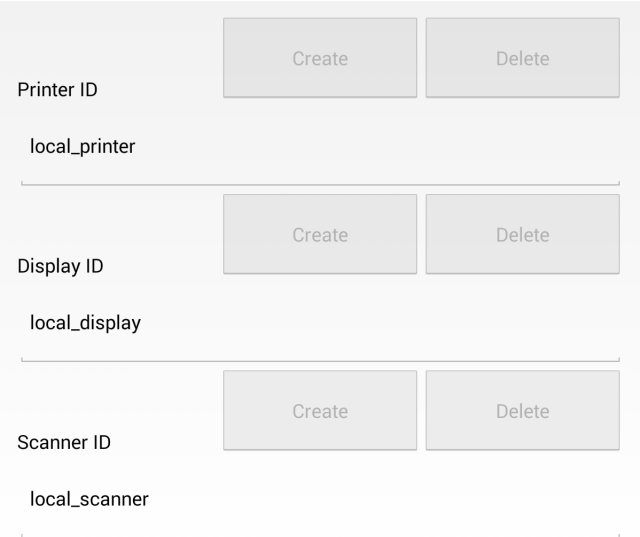
2. "ePOS-Device Sample Program for Android" screen appears. Click (Setting).

3. The screen shown below appears. Enter IP address of TM-i and click (Connect).



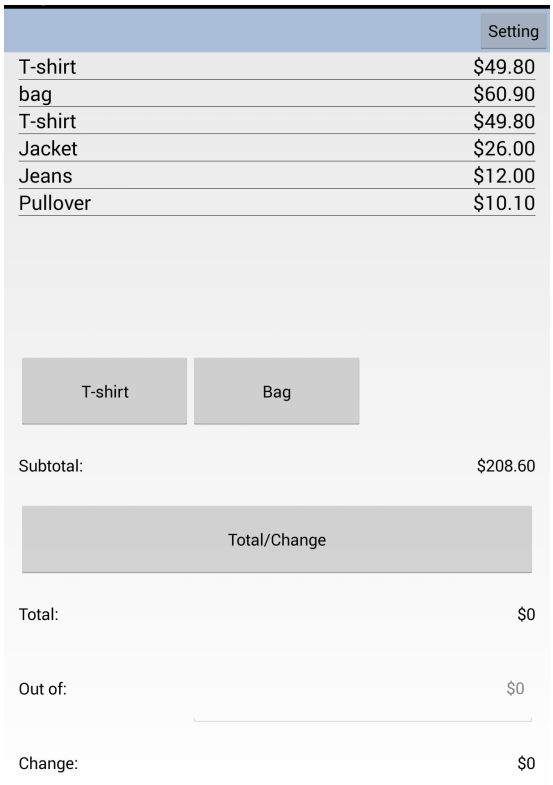
4 Create a device object following the next steps.

1. Enter "local_printer" in (Printer ID) and click (Connect).
2. Enter "local_display" in (Display ID) and click (Connect).
3. Enter "local_scanner" in (Scanner ID) and click (Connect).



5 Click (Back).

Operation



Print a receipt following the next steps.

1. Start the sample program. For details, refer to Environment Setting (p.159).
2. Set the sample program. For details, refer to Setting (p.162).
3. Register a product following the next steps.
   Once registered, the product and price are displayed in the customer display.
   - Click (T-shirt).
   - Click (Bag).
   - Read the barcode with a barcode scanner.

   Barcodes of products can be obtained in PDF of the sample program. Print and use it.
4. Click (Total/Change).
   The total price is displayed in the customer display.
5. Enter deposits in (Out of:).
6. Click (Total/Change).
   The change is displayed in the customer display and a receipt is printed.

Appendix

List of KeyCode

Constant	Code	Constant	Code	Constant	Code
Keyboard.VK_BACK	0x08	Keyboard.VK_7	0x37	Keyboard.VK_Z	0x5A
Keyboard.VK_TAB	0x09	Keyboard.VK_8	0x38	Keyboard.VK_MULTIPLY	0x6A
Keyboard.VK_RETURN	0x0D	Keyboard.VK_9	0x39	Keyboard.VK_ADD	0x6B
Keyboard.VK_SHIFT	0x10	Keyboard.VK_A	0x41	Keyboard.VK_SUBTRACT	0x6D
Keyboard.VK_CONTROL	0x11	Keyboard.VK_B	0x42	Keyboard.VK_F1	0x70
Keyboard.VK_MENU	0x12	Keyboard.VK_C	0x43	Keyboard.VK_F2	0x71
Keyboard.VK_CAPITAL	0x14	Keyboard.VK_D	0x44	Keyboard.VK_F3	0x72
Keyboard.VK_ESCAPE	0x1B	Keyboard.VK_E	0x45	Keyboard.VK_F4	0x73
Keyboard.VK_CONVERT	0x1C	Keyboard.VK_F	0x46	Keyboard.VK_F5	0x74
Keyboard.VK_NONCONVERT	0x1D	Keyboard.VK_G	0x47	Keyboard.VK_F6	0x75
Keyboard.VK_SPACE	0x20	Keyboard.VK_H	0x48	Keyboard.VK_F7	0x76
Keyboard.VK_PRIOR	0x21	Keyboard.VK_I	0x49	Keyboard.VK_F8	0x77
Keyboard.VK_NEXT	0x22	Keyboard.VK_J	0x4A	Keyboard.VK_F9	0x78
Keyboard.VK_END	0x23	Keyboard.VK_K	0x4B	Keyboard.VK_F10	0x79
Keyboard.VK_HOME	0x24	Keyboard.VK_L	0x4C	Keyboard.VK_F11	0x7A
Keyboard.VK_LEFT	0x25	Keyboard.VK_M	0x4D	Keyboard.VK_F12	0x7B
Keyboard.VK_UP	0x26	Keyboard.VK_N	0x4E	Keyboard.VK_OEM_1	0xBA
Keyboard.VK_RIGHT	0x27	Keyboard.VK_O	0x4F	Keyboard.VK_OEM_PLUS	0xBB
Keyboard.VK_DOWN	0x28	Keyboard.VK_P	0x50	Keyboard.VK_OEM_COMMA	0xBC
Keyboard.VK_INSERT	0x2D	Keyboard.VK_Q	0x51	Keyboard.VK_OEM_MINUS	0xBD
Keyboard.VK_DELETE	0x2E	Keyboard.VK_R	0x52	Keyboard.VK_OEM_PERIOD	0xBE
Keyboard.VK_0	0x30	Keyboard.VK_S	0x53	Keyboard.VK_OEM_2	0xBF
Keyboard.VK_1	0x31	Keyboard.VK_T	0x54	Keyboard.VK_OEM_3	0xC0
Keyboard.VK_2	0x32	Keyboard.VK_U	0x55	Keyboard.VK_OEM_4	0xDB
Keyboard.VK_3	0x33	Keyboard.VK_V	0x56	Keyboard.VK_OEM_5	0xDC
Keyboard.VK_4	0x34	Keyboard.VK_W	0x57	Keyboard.VK_OEM_6	0xDD
Keyboard.VK_5	0x35	Keyboard.VK_X	0x58	Keyboard.VK_OEM_7	0xDE
Keyboard.VK_6	0x36	Keyboard.VK_Y	0x59	Keyboard.VK_OEM_ATTN	0xF0

Printer Specifications

TM-T20II-i

		80 mm
Interface		Ethernet, Wireless LAN
Resolution		203 dpi x 203 dpi (W x H)
Print Width (42 Column Mode)		576 dots (546 dots)
Characters in a Line	Font A	ANK: 48 characters
	(42 Column Mode)	(ANK: 42 characters)
	Font B	ANK: 64 characters
	(42 Column Mode)	(ANK: 60 characters)
Character Size	Font A	ANK: 12 dots x 24 dots (W x H)
	(42 Column Mode)	(ANK: 13 dots x 24 dots)
	Font B	ANK: 9 dots x 17 dots (W x H)
	(42 Column Mode)	(ANK: 9 dots x 17 dots)
Character Baseline	Font A	At the 21st dot from the top of the character
	Font B	At the 16th dot from the top of the character
Default Line Feed Space		30 dots
Color Specification		First color
Page Mode Default Area		576 dots x 831 dots (W x H)
Page Mode Maximum Area		576 dots x 1662 dots (W x H)
Raster Image		Monochrome image
Logo		Monochrome image
Barcode		UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF,CODABAR, CODE93, CODE128, GS1-128, GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Limited, GS1 Databar Expanded
2D-Code		PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked, Composite Symbology
Ruled Line		Not supported

	80 mm
Paper Cut	Cut, Feed cut
Drawer Kick-Out	Supported
Buzzer	Optional
Paper Layout Settings	Not supported

TM-T70-i

		58 mm	80 mm
Interface		Ethernet, Wireless LAN	
Resolution		203 dpi x 203 dpi (W x H)	
Print Width		420 dots	576 dots
Characters in a Line	Font A	ANK: 34 characters	ANK: 48 characters
	Font B	ANK: 46 characters	ANK: 64 characters
Character Size	Font A	ANK: 12 dots x 24 dots (W x H)	
	Font B	ANK: 8 dots x 16 dots (W x H)	
Character Baseline	Font A	At the 21st dot from the top of the character	
	Font B	At the 15th dot from the top of the character	
Default Line Feed Space		30 dots	
Color Specification		First color	
Page Mode Default Area		576 dots x 1662 dots (W x H)	
Page Mode Maximum Area		576 dots x 1662 dots (W x H)	
Raster image		Monochrome image	
Logo		Monochrome image	
Barcode		UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128	
2D-Code		PDF417, QR Code	
Ruled Line		Not supported	
Paper Cut		Cut, Feed cut	
Drawer Kick-Out		Supported	
Buzzer		Not supported	
Paper Layout Settings		Not supported	

TM-T82II-i

		80 mm
Interface		Ethernet, Wireless LAN
Resolution		203 dpi x 203 dpi (W x H)
Print Width (42 Column Mode)		576 dots (546 dots)
Characters in a Line	Font A	ANK: 48 characters
	(42 Column Mode)	(ANK: 42 characters)
	Font B	ANK: 64 characters
	(42 Column Mode)	(ANK: 60 characters)
Character Size	Font A	ANK: 12 dots x 24 dots (W x H)
	(42 Column Mode)	(ANK: 13 dots x 24 dots)
	Font B	ANK: 9 dots x 17 dots (W x H)
	(42 Column Mode)	(ANK: 9 dots x 17 dots)
Character Baseline	Font A	At the 21st dot from the top of the character
	Font B	At the 16th dot from the top of the character
Default Line Feed Space		30 dots
Color Specification		First color
Page Mode Default Area		576 dots x 831 dots (W x H)
Page Mode Maximum Area		576 dots x 1662 dots (W x H)
Raster image		Monochrome image
Logo		Monochrome image
Barcode		UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF,CODABAR, CODE93, CODE128, GS1-128, GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Limited, GS1 Databar Expanded
2D-Code		PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked, Composite Symbology
Ruled Line		Not supported
Paper Cut		Cut, Feed cut
Drawer Kick-Out		Supported
Buzzer		Optional
Paper Layout Settings		Not supported

TM-T83II-i/ TM-T83II

		80 mm
Interface		Ethernet, Wireless LAN
Resolution		180 dpi x 180 dpi (W x H)
Print Width		512 dots
Characters in a Line	Font A	ANK: 42 characters Kanji: 21 characters
	Font B	ANK: 56 characters Kanji: 32 characters
Character Size	Font A	ANK: 12 dots x 24 dots (W x H) Kanji: 24 dots x 24 dots (W x H)
	Font B	ANK: 9 dots x 17 dots (W x H) Kanji: 16 dots x 16 dots (W x H)
Character Baseline	Font A	ANK: At the 21st dot from the top of the character Kanji: At the 21st dot from the top of the character
	Font B	ANK: At the 16th dot from the top of the character Kanji: At the 15th dot from the top of the character
Default Line Feed Space		30 dots
Color Specification		First color
Page Mode Default Area		512 dots x 1662 dots (W x H)
Page Mode Maximum Area		512 dots x 1662 dots (W x H)
Raster image		Monochrome image
Logo		Monochrome image
Barcode		UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128, GS1-128, GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Limited, GS1 DataBar Expanded
2D-Code		PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked
Ruled Line		Not supported
Page Mode	Line	Not supported
	Rectangle	

	80 mm
Paper Cut	Cut, Feed cut
Drawer Kick-Out	Supported
Buzzer	Optional
Paper Layout Settings	Not supported

TM-T88V-i/ TM-T88V

* Only for South Asia model

		58 mm	80 mm
Interface		Ethernet, Wireless LAN	
Resolution		180 dpi x 180 dpi (W x H)	
Print Width		360 dots	512 dots
Characters in a Line	Font A	ANK: 30 characters	ANK: 42 characters
	Font B	ANK: 40 characters	ANK: 56 characters
	Special font A*	30 characters	42 characters
	Special font B*	40 characters	56 characters
Character Size	Font A	ANK: 12 dots x 24 dots (W x H)	
	Font B	ANK: 9 dots x 17 dots (W x H)	
	Special font A*	12 dots x 24 dots (W x H)	
	Special font B*	9 dots x 24 dots (W x H)	
Character Baseline	Font A	At the 21st dot from the top of the character	
	Font B	At the 16th dot from the top of the character	
	Special font A*	At the 20th dot from the top of the character	
	Special font B*	At the 20th dot from the top of the character	
Default Line Feed Space		30 dots	
Color Specification		First color	
Page Mode Default Area		360 dots x 831 dots (W x H)	512 dots x 831 dots (W x H)
Page Mode Maximum Area		360 dots x 1662 dots (W x H)	512 dots x 1662 dots (W x H)
Raster Image		Monochrome image, Gray scale image	
Logo		Monochrome image, Gray scale image	
Barcode		UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128, GS1-128,GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Limited, GS1 Databar Expanded	

* Only for South Asia model

	58 mm	80 mm
2D-Code	PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked (Composite Symbology not supported)	
Ruled Line	Not supported	
Paper Cut	Cut, Feed cut	
Drawer Kick-Out	Supported	
Buzzer	Optional	
Paper Layout Settings	Not supported	

TM-L90

		Receipt	Die-cut Label
Interface		Ethernet, Wireless LAN	
Resolution		203 dpi x 203 dpi (W x H)	
Print Width		256 dots (38 mm) to 576 dots (80 mm)	224 dots (38 mm) to 560 dots (80 mm)
Font		For more information about what character codes can be printed, refer to the user's manual that came with the printer.	
Characters in a Line	Font A	ANK: 48 characters,	ANK: 46 characters
	Font B	ANK: 570 characters	ANK: 56 characters
Character Size	Font A	ANK: 12 dots x 24 dots (W x H)	
	Font B	ANK: 10 dots x 24 dots (W x H)	
Character Baseline	Font A	At the 21st dot from the top of the character	
	Font B	At the 21st dot from the top of the character	
Default Line Feed Space		30 dots	
Color Specification		First color Second color, Second color (when two-color printing is set)	
Page Mode Default Area		576 dots x 738 dots (W x H)	560 dots x 738 dots (W x H)
	when two-color printing is set	576 dots x 369 dots (W x H)	560 dots x 369 dots (W x H)
Page Mode Maximum Area		576 dots x 1476 dots (W x H)	560 dots x 1476 dots (W x H)
	when two-color printing is set	576 dots x 738 dots (W x H)	560 dots x 738 dots (W x H)
Raster Image		Monochrome image, Two color image	
Logo		Monochrome image, Two color image (To perform two-color printing, change the settings of the printer using the memory switch setting utility.)	
Barcode		UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128	
2D-Code		PDF417, QR Code, MaxiCode	
Ruled Line		Not supported	
Paper Cut		Cut, Feed cut	

	Receipt	Die-cut Label
Drawer Kick-Out	Supported	
Buzzer	Not supported	
Paper Layout Settings	Not supported (With automatic setup mode)	

TM-P20

		58 mm
Interface		Ethernet, Wi-Fi
Resolution		203 dpi x 203 dpi (W x H)
Print Width		384 dots
Characters in a Line	Font A	ANK: 32 characters
	Font B	ANK: 42 characters
	Font C	ANK: 42 characters
	Font D	ANK: 38 characters
	Font E	ANK: 48 characters
Character Size	Font A	ANK: 12 dots x 24 dots (W x H)
	Font B	ANK: 9 dots x 24 dots (W x H)
	Font C	ANK: 9 dots x 17 dots (W x H)
	Font D	ANK: 10 dots x 24 dots (W x H)
	Font E	ANK: 8 dots x 16 dots (W x H)
Character Baseline	Font A	At the 21st dot from the top of the character
	Font B	At the 16th dot from the top of the character
	Font C	At the 16th dot from the top of the character
	Font D	At the 21st dot from the top of the character
	Font E	At the 15th dot from the top of the character
Default Line Feed Space		30 dots
Color Specification		First color
Page Mode Default Area		384 dots x 2400 dots (W x H)
Page Mode Maximum Area		384 dots x 2400 dots (W x H)
Raster Image		Monochrome image
Logo		Monochrome image
Barcode		UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF,CODABAR, CODE93, CODE128, GS1-128, GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Limited, GS1 DataBar Expanded

	58 mm
2D-Code	PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked, Composite Symbology
Ruled Line	Not supported
Paper Cut	Feeds paper to cutting position
Drawer Kick-Out	Not supported
Buzzer	Optional
Paper Layout Settings	Supported

TM-P60II

		Receipt 58 mm	Receipt 60 mm
Interface		Wireless LAN	
Resolution		203 dpi x 203 dpi (W x H)	
Print Width		420 dots	432 dots
Characters in a Line	Font A	ANK: 35 characters,	ANK: 36 characters
	Font B	ANK: 42 characters	ANK: 43 characters
	Font C	ANK: 52 characters	ANK: 54 characters
Character Size	Font A	ANK: 12 dots x 24 dots (W x H)	
	Font B	ANK: 10 dots x 24 dots (W x H)	
	Font C	ANK: 8 dots x 16 dots (W x H)	
Character Baseline	Font A	At the 21st dot from the top of the character	
	Font B	At the 21st dot from the top of the character	
	Font C	At the 15th dot from the top of the character	
Default Line Feed Space		30 dots	
Color Specification		First color	
Page Mode Default Area		420 dots x 1624 dots (W x H)	432 dots x 1624 dots (W x H)
Page Mode Maximum Area		420 dots x 1624 dots (W x H)	432 dots x 1624 dots (W x H)
Raster Image		Monochrome image	
Logo		Monochrome image	
Barcode		UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128, GS1-128, GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Limited, GS1 DataBar Expanded	
2D-Code		PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked, Aztec Code, DataMatrix (Composit Symbology : Not supported)	
Ruled Line		Not supported	
Paper Cut		Cut, Feed cut	
Drawer Kick-Out		Not supported	
Buzzer		Support (Pattern 1 ~ Pattern 10, Stop)	
Paper Layout Settings		Not supported	

TM-P60II with Peeler

		Receipt 58 mm	Receipt 60 mm	Die-cut label
Interface		Wireless LAN		
Resolution		203 dpi x 203 dpi (W x H)		
Print Width		420 dots	432 dots	160 dots ~ 400 dots
Characters in a Line	Font A	ANK: 35 characters,	ANK: 36 characters	ANK: 33 characters
	Font B	ANK: 42 characters	ANK: 43 characters	ANK: 40 characters
	Font C	ANK: 52 characters	ANK: 54 characters	ANK: 50 characters
Character Size	Font A	ANK: 12 dots x 24 dots (W x H)		
	Font B	ANK: 10 dots x 24 dots (W x H)		
	Font C	ANK: 8 dots x 16 dots (W x H)		
Character Baseline	Font A	At the 21st dot from the top of the character		
	Font B	At the 21st dot from the top of the character		
	Font C	At the 15th dot from the top of the character		
Default Line Feed Space		30 dots		
Color Specification		First color		
Page Mode Default Area		420 dots x 1624 dots (W x H)	432 dots x 1624 dots (W x H)	400 dots x 1624 dots (W x H)
Page Mode Maximum Area		420 dots x 1624 dots (W x H)	432 dots x 1624 dots (W x H)	400 dots x 1624 dots (W x H)
Raster Image		Monochrome image		
Logo		Monochrome image		
Barcode		UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128, GS1-128, GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Limited, GS1 DataBar Expanded		
2D-Code		PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked, Aztec Code, DataMatrix (Composit Symbology : Not supported)		
Ruled Line		Not supported		
Paper Cut		Feed cut (Feeds paper to cutting position)		
Drawer Kick-Out		Not supported		
Buzzer		Support (Pattern 1 ~ Pattern 10, Stop)		
Paper Layout Settings		Supported		

TM-P80

		80 mm
Interface		Wireless LAN
Resolution		203 dpi x 203 dpi (W x H)
Print Width		576 dots
Characters in a Line	Font A	ANK: 48 characters
	Font B	ANK: 64 characters
Character Size	Font A	ANK: 12 dots x 24 dots (W x H)
	Font B	ANK: 9 dots x 17 dots (W x H)
Character Baseline	Font A	At the 21st dot from the top of the character
	Font B	At the 16th dot from the top of the character
Default Line Feed Space		30 dots
Color Specification		First color
Page Mode Default Area		576 dots x 1662 dots (W x H)
Page Mode Maximum Area		576 dots x 1662 dots (W x H)
Raster Image		Monochrome image
Logo		Monochrome image
Barcode		UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF,CODABAR, CODE93, CODE128, GS1-128, GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Limited, GS1 Databar Expanded
2D-Code		PDF417, QR Code, MaxiCode, Data Matrix, Aztec Code, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked, Composite Symbology
Ruled Line		Not supported
Paper Cut		Feed
Drawer Kick-Out		Not supported
Buzzer		Optional
Paper Layout Settings		Supported

TM-T20

		58 mm	80 mm
Interface		Ethernet	
Resolution		203 dpi x 203 dpi (W x H)	
Print Width		420 dots	576 dots
Characters in a Line	Font A	ANK: 35 characters,	ANK: 48 characters
	Font B	ANK: 46 characters	ANK: 64 characters
Character Size	Font A	ANK: 12 dots x 24 dots (W x H)	
	Font B	ANK: 9 dots x 17 dots (W x H)	
Character Baseline	Font A	At the 21st dot from the top of the character	
	Font B	At the 16th dot from the top of the character	
Default Line Feed Space		30 dots	
Color Specification		First color	
Page Mode Default Area		420 dots x 831 dots (W x H)	576 dots x 831 dots (W x H)
Page Mode Maximum Area		420 dots x 1662 dots (W x H)	576 dots x 1662 dots (W x H)
Raster Image		Monochrome image	
Logo		Monochrome image	
Barcode		UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128, GS1-128, GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Limited, GS1 DataBar Expanded	
2D-Code		PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked (Composit Symbology : Not supported)	
Ruled Line		Not supported	
Paper Cut		Cut, Feed cut	
Drawer Kick-Out		Supported	
Buzzer		Optional (Pattern A ~ Pattern E, Error, No paper, Stop)	
Paper Layout Settings		Not supported	

TM-T20II

		58 mm	80 mm
Interface		Ethernet, Wireless LAN	
Resolution		203 dpi x 203 dpi (W x H)	
Print Width (42 Column Mode)		420 dots (378 dots)	576 dots (546 dots)
Characters in a Line (42 Column Mode)	Font A	ANK: 35 characters (ANK: 42 characters)	ANK: 48 characters (ANK: 42 characters)
	Font B	ANK: 46 characters (ANK: 31 characters)	ANK: 64 characters (ANK: 60 characters)
Character Size (42 Column Mode)	Font A	ANK: 12 dots x 24 dots (W x H)	
		(ANK: 9 dots x 17 dots)	(ANK: 13 dots x 24 dots)
	Font B	ANK: 9 dots x 17 dots (W x H)	
		(ANK: 12 dots x 24 dots)	(ANK: 9 dots x 17 dots)
Character Baseline	Font A	At the 21st dot from the top of the character	
	Font B	At the 16th dot from the top of the character	
Default Line Feed Space		30 dots	
Color Specification		First color	
Page Mode Default Area		420 dots x 831 dots (W x H)	576 dots x 831 dots (W x H)
Page Mode Maximum Area		420 dots x 1662 dots (W x H)	576 dots x 1662 dots (W x H)
Raster Image		Monochrome image	
Logo		Monochrome image	
Barcode		UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF,CODABAR, CODE93, CODE128, GS1-128, GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Limited, GS1 Databar Expanded	
2D-Code		PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked, Composite Symbology	
Ruled Line		Not supported	
Paper Cut		Cut, Feed cut	
Drawer Kick-Out		Supported	
Buzzer		Optional	
Paper Layout Settings		Not supported	

TM-T70

		80 mm
Interface		Ethernet, Wireless LAN
Resolution		180 dpi x 180 dpi (W x H)
Print Width		512 dots
Characters in a Line	Font A	ANK: 42 characters
	Font B	ANK: 56 characters
Character Size	Font A	ANK: 12 dots x 24 dots (W x H)
	Font B	ANK: 9 dots x 17 dots (W x H)
Character Baseline	Font A	At the 21st dot from the top of the character
	Font B	At the 15th dot from the top of the character
Default Line Feed Space		30 dots
Color Specification		First color
Page Mode Default Area		512 dots x 1662 dots (W x H)
Page Mode Maximum Area		512 dots x 1662 dots (W x H)
Raster image		Monochrome image
Logo		Monochrome image
Barcode		UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128
2D-Code		PDF417, QR Code
Ruled Line		Not supported
Paper Cut		Cut, Feed cut
Drawer Kick-Out		Supported
Buzzer		Not supported
Paper Layout Settings		Not supported

TM-T70II

		80 mm
Interface		Ethernet, Wireless LAN
Resolution		203 dpi x 203 dpi (W x H)
Print Width		576 dots
Characters in a Line	Font A	ANK: 48 characters
	Font B	ANK: 72 characters
Character Size	Font A	ANK: 12 dots x 24 dots (W x H)
	Font B	ANK: 9 dots x 17 dots (W x H)
Character Baseline	Font A	At the 21st dot from the top of the character
	Font B	At the 15th dot from the top of the character
Default Line Feed Space		30 dots
Color Specification		First color
Page Mode Default Area		576 dots x 1662 dots (W x H)
Page Mode Maximum Area		576 dots x 1662 dots (W x H)
Raster Image		Monochrome image
Logo		Monochrome image
Barcode		UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128, GS1-128,GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Limited, GS1 Databar Expanded
2D-Code		PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked
Ruled Line		Not supported
Paper Cut		Cut, Feed cut
Drawer Kick-Out		Supported
Buzzer		Supported
Paper Layout Settings		Not supported

TM-T82II

		58 mm	80 mm
Interface		Ethernet	
Resolution		203 dpi x 203 dpi (W x H)	
Print Width		420 dots	576 dots
Characters in a Line	Font A	ANK: 35 characters	ANK: 48 characters
	Font B	ANK: 46 characters	ANK: 64 characters
Character Size	Font A	ANK: 12 dots x 24 dots (W x H)	
	Font B	ANK: 9 dots x 17 dots (W x H)	
Character Baseline	Font A	At the 21st dot from the top of the character	
	Font B	At the 16th dot from the top of the character	
Default Line Feed Space		30 dots	
Color Specification		First color	
Page Mode Default Area		420 dots x 831 dots (W x H)	576 dots x 831 dots (W x H)
Page Mode Maximum Area		420 dots x 1662 dots (W x H)	576 dots x 1662 dots (W x H)
Raster Image		Monochrome image	
Logo		Monochrome image	
Barcode		UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF,CODABAR, CODE93, CODE128, GS1-128, GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Limited, GS1 Databar Expanded	
2D-Code		PDF417, QR Code, MaxiCode, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, GS1 DataBar Expanded Stacked, Composite Symbology	
Ruled Line		Not supported	
Paper Cut		Cut, Feed cut	
Drawer Kick-Out		Supported	
Buzzer		Optional	
Paper Layout Settings		Not supported	

TM-T88IV

		80 mm	58 mm
Interface		Ethernet, Wireless LAN	
Resolution		180 dpi x 180 dpi (W x H)	
Print Width		512 dots	360 dots
Characters in a Line	Font A	ANK: 42 characters	ANK: 30 characters
	Font B	ANK: 56 characters	ANK: 40 characters
Character Size	Font A	ANK: 12 dots x 24 dots (W x H)	
	Font B	ANK: 9 dots x 17 dots (W x H)	
Character Baseline	Font A	At the 21st dot from the top of the character	
	Font B	At the 16th dot from the top of the character	
Default Line Feed Space		30 dots	
Color Specification		First color First color, Second color (when two-color printing is set)	
Page Mode Default Area		512 dots x 831 dots (W x H)	360 dots x 831 dots (W x H)
	when two-color printing is set	512 dots x 415 dots (W x H)	360 dots x 415 dots (W x H)
Page Mode Maximum Area		512 dots x 1662 dots (W x H)	360 dots x 1662 dots (W x H)
	when two-color printing is set	512 dots x 831 dots (W x H)	360 dots x 831 dots (W x H)
Raster image		Monochrome image, two-color image	
Logo		Monochrome image, two-color image (To perform two-color printing, change the settings of the printer using the memory switch setting utility.)	
Bar Code		UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128, GS1-128	
2D-Code		PDF417, QR Code	
Ruled Line		Not supported	
Paper Cut		Cut, Feed cut	
Drawer Kick-Out		Supported	
Buzzer		Not supported	
Paper Layout Settings		Not supported	

TM-T90

		58 mm	60 mm	80 mm
Interface		Ethernet, Wireless LAN		
Resolution		180 dpi x 180 dpi (W x H)		
Print Width		360 dots	384 dots	512 dots
Characters in a Line	Font A	ANK: 30 characters,	ANK: 32 characters	ANK: 42 characters
	Font B	ANK: 40 characters	ANK: 42 characters	ANK: 56 characters
Character Size	Font A	ANK: 12 dots x 24 dots (W x H)		
	Font B	ANK: 9 dots x 17 dots (W x H)		
Character Baseline	Font A	At the 21st dot from the top of the character		
	Font B	At the 16th dot from the top of the character		
Default Line Feed Space		30 dots		
Color Specification		First color Second color, Second color (when two-color printing is set)		
Page Mode Default Area		360 dots x 831 dots (W x H)	384 dots x 831 dots (W x H)	512 dots x 831 dots (W x H)
	when two-color printing is set	360 dots x 415 dots (W x H)	384 dots x 415 dots (W x H)	512 dots x 415 dots (W x H)
Page Mode Maximum Area		360 dots x 1662 dots (W x H)	384 dots x 1662 dots (W x H)	512 dots x 1662 dots (W x H)
	when two-color printing is set	360 dots x 831 dots (W x H)	384 dots x 831 dots (W x H)	512 dots x 831 dots (W x H)
Raster Image		Monochrome image, Two color image		
Logo		Monochrome image, Two color image (To perform two-color printing, change the settings of the printer using the memory switch setting utility.)		
Barcode		UPC-A, UPC-E, EAN13, JAN13, EAN8, JAN8, CODE39, ITF, CODABAR, CODE93, CODE128		
2D-Code		PDF417		
Ruled Line		Not supported		
Paper Cut		Cut, Feed cut		
Drawer Kick-Out		Supported		

	58 mm	60 mm	80 mm
Buzzer	Supported via Drawer Kick-Out		
Paper Layout Settings	Not supported		

TM-U220

		76 mm	70 mm	58 mm
Interface		Ethernet, Wireless LAN		
Resolution		80 dpi x 72 dpi (W x H)		
Print Width		200 dots	180 dots	150 dots
Characters in a Line	Font A	ANK: 33 characters,	ANK: 30 characters	ANK: 25 characters
	Font B	ANK: 40 characters	ANK: 36 characters	ANK: 30 characters
Character Size	Font A	ANK: 4.5 dots x 9 dots (W x H)		
	Font B	ANK: 3.5 dots x 9 dots (W x H)		
Character Baseline	Font A	Bottom of the characters		
	Font B	Bottom of the characters		
Default Line Feed Space		12 dots		
Color Specification		First color First color, Second color (When using a two-color ribbon cassette)		
Page Mode Default Area		Not supported		
Page Mode Maximum Area		Not supported		
Raster Image		Monochrome image		
Logo		Not supported		
Barcode		Not supported		
2D-Code		Not supported		
Ruled Line		Not supported		
Paper Cut		Cut, Feed cut		
Drawer Kick-Out		Supported		
Buzzer		Not supported		
Paper Layout Settings		Not supported		

Open Source License

This product contains the following open-source libraries:

libxml2

Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar licence but with different Copyright notices) all the files are:

Copyright (C) 1998-2003 Daniel Veillard. All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

icu4c

Copyright (C) 2008 The Android Open Source Project

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and limitations under the License.